

AI BOTS • CODING FOR BOTS • DIGILENT'S BOT

SERVO

FOR THE ROBOT INNOVATOR

www.servomagazine.com

MAGAZINE

January 2010

Nuts, Bolts & Thingamajigs

Nurturing The Tinkering Spirit



◆ **Simple Sensor Interface**

◆ **Robot Optics Using A Webcam**

◆ **Tips For Selecting DC Motors**

U.S. \$5.50 CANADA \$7.00



The Evolution Continues

BIOLOID

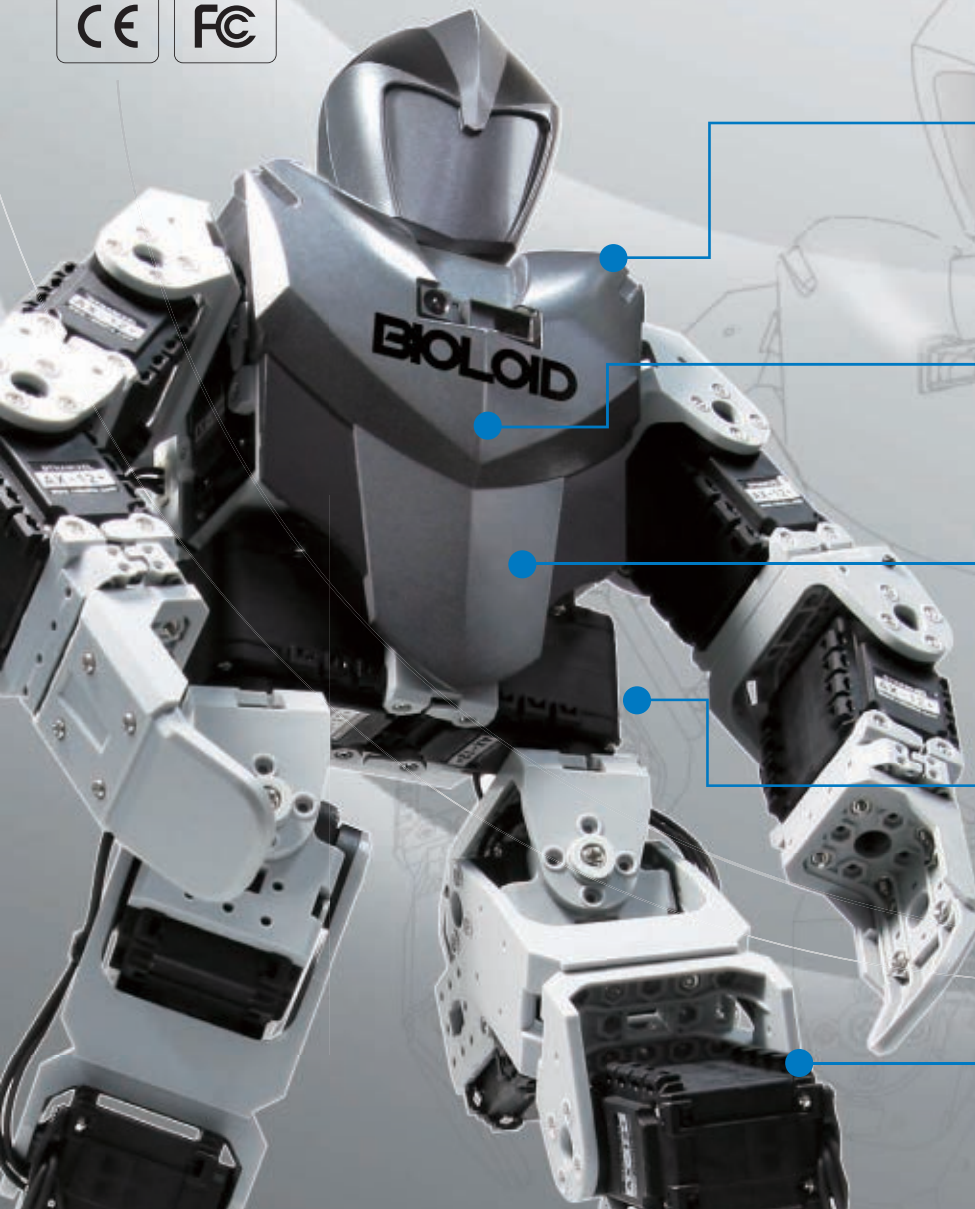
PREMIUM Kit

New Release
Special Gift



Zigbee module
for 1,000 customers

All-in-one package for
advanced robot builders



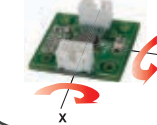
CM-510
Main Controller



RoboPlus



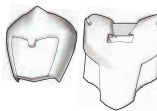
Gyro Sensor
Walk Balancing



DMS
Distance Measurement



Humanoid Skin



RC-100
Remote Controller



Li-Po Battery
11.1V 1000mA



USB2Dynamixel



Dynamixel AX-12+
18 DOF Actuators



IR Sensor
Object Detection



- Excellent humanoid walking performance (Adjusting posture while walking)
- Various sensors including Gyro, Distance, IR and more external ports
- Remote control capability (IR-default, Zigbee-optional)
- C-style programming & motion teaching with RoboPlus S/W (USB interface included)
- Transparent humanoid skin for customization
- Digital Packet communication with daisy chain topology
- Building various robots using versatile expansion mechanism

ROBOTIS

www.robotis.com

TEL : +82-70-8671-2600

Email : contactus2@robotis.com



Robots, lots of robots!

The World's Leading Source
for Domestic and Professional Robot Technology



www.robotshop.com



PAGE 12

Departments

- | | | | |
|-----------|---------------------------------------|-----------|-----------------|
| 06 | Mind/Iron | 67 | SERVO Webstore |
| 08 | Bio-Feedback | 70 | Events Calendar |
| 20 | Bots in Brief | 82 | Robo-Links |
| 26 | New Products | 82 | Advertiser's |
| 51 | Showcase | | Index |
| 24 | <i>Special New Product Commentary</i> | | |

The Combat Zone...

Features

- 28** PARTS IS PARTS:
My First Welder (or not?)
- 29** MANUFACTURING:
Debugging Welding
Problems

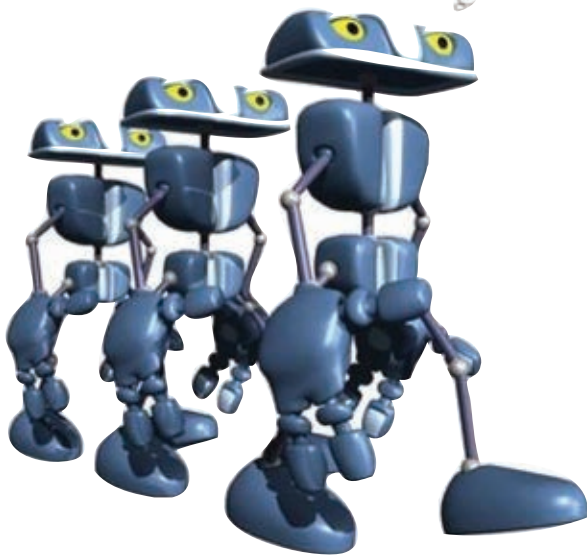
Events

- 30** Results/Upcoming Events
- 31** EVENT REPORT:
No Gain Without Pain —
Franklin Institute 2009

Columns

- 09** Robytes
by Jeff Eckert
Stimulating Robot Tidbits
- 12** GeerHead
by David Geer
*Artificial Intelligence Brings
Humanoid Robots to Life*
- 15** Ask Mr. Roboto
by Dennis Clark
Your Problems Solved Here
- 71** Twin Tweaks
by Bryce and Evan Woolley
The Education of Budding Roboticians
- 77** Then and Now
by Tom Carroll
The Future of Robotics Competitions

In This Issue ...



33 Tips for Selecting DC Motors for Your Mobile Robot

by A.J. Neal

When building a mobile robot, selecting the drive motors is one of the most important decisions you will make.

38 Optics for Your Robot Using a Webcam

by Mike Keesling

Learn some basics about lenses, light, and color to get you started using machine vision.

42 From Autonomy to Symbiosis: Some Thoughts on the Complexities of Networked Robots

by Fulvio Mastrogiovanni

Certain actions are much easier for humans than robots, so, how will they ever make it to the workplace with us?

46 Digilent's Robotic Starter Kit

by Fred Eady

Once we sort through this new kit, we'll get a joystick, three momentary pushbuttons, and a pair of LEDs up and running.

52 Self Similar Optimization

by Sean O'Connor

This new coding idea will change how you design and optimize robots.

55 Simple Sensor Interface

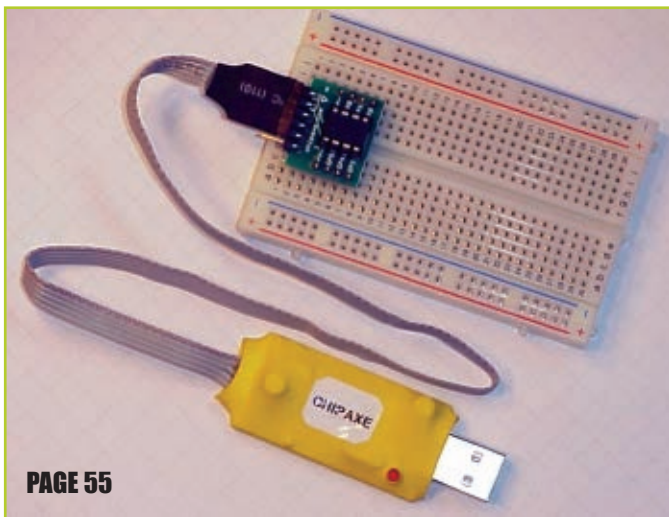
by the CHIPAXE Team

See how to implement this new chip on the block into some extremely simple and low cost sensor applications.

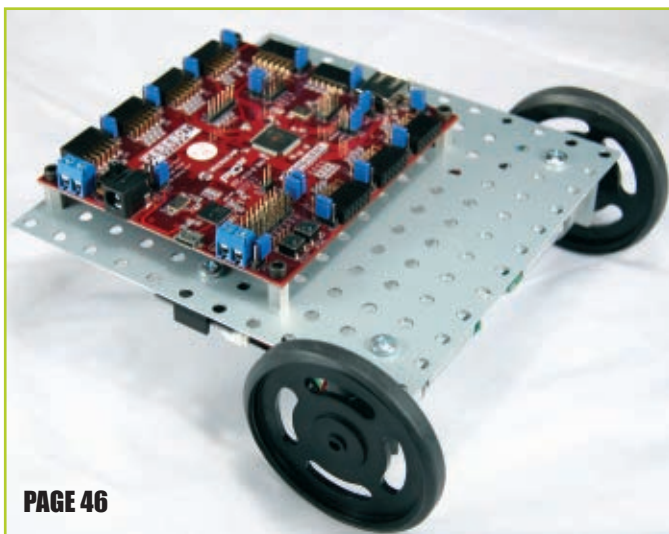
60 Hacker Dojos Offer Collaboration Opps for Robot Hobbyists

by Jeremy Kerfs

Developer communities help everyone. See what came out of a recent "Random Hacks of Kindness" event at a Hacker Dojo in Mountain View, CA.



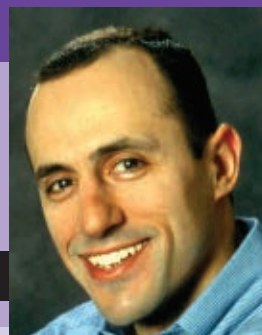
PAGE 55



PAGE 46

Mind / Iron

by Bryan Bergeron, Editor



Little Hands Build Big Dreams

If you're a robotics enthusiast, you don't need to be convinced of the importance of developing mechanical skills, including a hands-on familiarity with various materials, hardware components, and assembly techniques. In robotics, fluency in basic programming and mechanical engineering principles is necessary but insufficient to design and develop anything new.

Consider that robotic control algorithms that work on simulated robots often don't run on the real thing because of real-world issues such as friction, imperfect motor startup characteristics, and other factors not typically represented in the simulation — that is, unless the programmer has experience with the real thing. In robotics, there's no substitute for a physical platform to build on and test real components, and to develop real world skills.

As a kid growing up in a town on the gulf coast that serviced the offshore industry, I assumed that everyone on the planet owned or had access to a metal lathe, welding machine, milling machine, and master mechanics tool set. Today, living in Boston, I happen to own a miniature milling machine and fairly extensive tool collection, but I'm certain my neighbors don't. In fact, given that Boston is a typical college town, machining and other blue-collar skills are not highly regarded. After all, technicians and factory workers build things with their hands, engineers design things, and physicists theorize what's possible. In short, hands-on, blue-collar skills are not what we aspire to, for ourselves or for our children.

Transition to our front cover, featuring John Ratzenberger, perhaps best known for his roles as Cliff, the postman in *Cheers*, and the host of the Travel Channel show *John Ratzenberger's Made in America*. I had the pleasure of speaking with John about his charity organization Nuts, Bolts, and Thingamajigs Foundation (NBT), which is dedicated to inspiring the next generation of welders, plumbers, carpenters, and others who work with their hands and minds, one tinker at a time. John is convinced that the backbone of civilization is manufacturing, and the future belongs to those who know how to use their hands.

John's conviction stems from both firsthand

knowledge and those intimately familiar with the American workforce. John worked as a journeyman carpenter throughout New England prior to becoming an actor. He started NBT about 2-1/2 years ago following his work with the *Made in America* show. During his travels for the show, he frequently encountered CEOs alarmed that the US is running out of people who can build and make things with their hands.

As an actor, John is attuned to the role of the general public's attitude towards the loss of manufacturing jobs in the US. He pointed out that in movies, those who work with their hands are portrayed as geeks, but they're really the heroes. As such, one of his goals is to improve the image of those who work with their hands to make things. And he's going about it by taking strategic, direct action.

For example, guidance counselors often steer students away from careers in manufacturing because they perceive the jobs as low paying, with grimy work conditions reminiscent of the factories portrayed in 1930's movies. To confront this misperception head on, NBT took busloads of counselors on tours of manufacturing plants. They were shown that the plants are clean, that the workers make a good living, and that most have nice homes. As a result, these counselors now encourage students to consider careers that involve mechanical skills.

About Nuts, Bolts, and Thingamajigs

Nuts, Bolts, and Thingamajigs encourages the next generation of engineers, carpenters, welders, plumbers, mechanics, builders, manufacturers, electricians — manual artists to pursue careers in manufacturing. NBT works to dispel the myth that a career in manufacturing is a dirty, low paying job, and dominated by men. Most of Nuts and Bolts' summer camps celebrate the growing number of girls attending. NBT also provides training programs; scholarships to one, two, and four year technical and vocational schools, and works in Washington, DC to bring attention to the decline of skilled labor in America.

John's quick to point out that more has to be done in the American educational system. For example, shop classes — once popular — are now rare. As a result, most students who finish high school are functionally illiterate when it comes to basic mechanical skills such as the ability to read a ruler. He also related a complaint that many employers have of new engineering graduates. Apparently, they often can't build anything because they don't know how things are made. Theoretical knowledge alone just doesn't cut it on the shop floor. To compound the problem, there's a shortage of engineers. According to John, American universities awarded twice as many

Fast Facts

The average age of a manufacturing employee in America is 56 years old.

America's youth lacks basic skills in math, science, and even the ability to read a ruler.

25% of American kids don't graduate high school. In some states, it's as high as 45%.

Source — American Welding Society:

By 2010, demand for skilled welders may outstrip supply by about 200,000. Part of the problem is retiring baby boomers — in 2009, the average age of welders was 54 and it keeps climbing.

The ranks of welders, brazers, and solderers — (those whose job it is to join pieces of metal) has dropped 10 percent since 2000, according to the federal Bureau of Labor Statistics.

Source — Center for Workforce Development:

Manufacturing is the third largest major industry sector in the US (behind retail trade and health care), and contributes 14.1 million jobs to the economy.

Nationwide, 4.3 percent of firms fall into the manufacturing sector, but account for 12.5 percent of employment and 15.2 percent of wages.

Within the 86 major subcategories of the manufacturing sector, motor vehicle parts manufacturing accounts for the largest share of employment (4.6 percent), followed by plastics product manufacturing (4.5 percent), printing (4.5 percent), animal processing (3.6 percent), aerospace manufacturing (3.3 percent), and semiconductor manufacturing (3.2 percent).

The United States still holds the greatest shares of the world's GDP.

degrees in sports management than in engineering last year. John Ratzenberger certainly has his work cut out for him, and, of course, he can't do it alone. Fortunately, Nuts and Bolts recently merged with the Fabricators & Manufacturers Association of America, which works on multi-level platforms to promote American manufacturing, including grants and scholarships to nonprofits that provide day or overnight camps to children who want to learn the manual arts. To learn more about John and Nuts, Bolts, and Thingamajigs, and how you too can help, log onto www.nutsandboltsfoundation.org.

Through his charity (Nuts, Bolts, and Thingamajigs Foundation), John has committed his resources to introducing America's youth to the pleasures of 'tinkering' — getting away from their video games and TV sets and into the backyard building things. In that way, we will create the next generation of artisans, inventors, engineers, repairmen, and skilled workers — in short, a self-sufficient, self-sustaining society. John's tag line has become "Little hands build big dreams. Give children tools and watch them build America."

John is an outspoken advocate for American-made products and the companies that keep Americans working. In 2007, John embarked on a yearlong commitment with the Association for American Manufacturing and US Steelworkers to create a Presidential Town Hall Tour. The Town Hall series brought attention to issues that American voters were demanding to hear about — a real commitment from presidential candidates to ensure a strong manufacturing industry. During the town hall events, John encouraged voters to ask the presidential candidates what specific policies they would enact to strengthen the American manufacturing base, which is vital to our economic and national security.

John was invited to address Congress and its Manufacturing Caucus that same year, for which he prepared his oft-quoted speech "The Industrial Tsunami Heading Our Way." He continues to work with politicians on both sides of the aisle to ensure that the American manufacturing industry has a voice in Washington.

During his free time, John is an avid sailor, fisherman, and billiard player. He enjoys international travel, fencing, and collecting antiques. He plays the drums and belongs to a bagpipe band, as part of the Emerald Society. Sports such as karate, yoga, and skeet shooting keep him active. He has one son and one daughter and lives outside of Los Angeles, but spends as much time as possible on his boat, cruising up and down the east coast. **SV**

BIO-FEEDBACK

I recently participated in the BEST Robotics program in Oklahoma. My team placed seventh out of 12 teams. This was our first year to participate and my team learned a great deal from the experience. Just a few of the things we learned were cooperation, planning and time-management skills, dividing tasks, engineering techniques, and the proper and safe use of shop tools.

Thank you, *SERVO*, for your contributions to the BEST Robotics program. Without your generous support, this program would not be possible.

Silas Percival
Apache, OK

It's disappointing to see so many of the people shown in the "Taking a Walk on the Wild Side" article (November '09) working without eye protection. You can replace a broken robot component, but not damaged eyesight. Work safely.

Jon Titus
Herriman, UT

It might be late to point this out, but 2009 marks the 20th anniversary of *Fast, Cheap, and Out of Control: A Robot Invasion of The Solar System*. I thought that this should not go unnoticed. After all, look at what it influenced:

- An award winning documentary

of the same name.

- Amateur robotics. (Didn't you want to see Attila or Ghengis under the Christmas Tree?)

- BEAM robotics. (Transistors and discrete logic chips at that time cost less than microcontrollers. And no programming = no control.)

Oddly enough, the only people not influenced were the target audience: NASA. True, they did have the motto "better, faster, cheaper" during Dan Goldin's administration. And Pathfinder — when adjusted for inflation — was cheaper than Viking. But it was hardly out of control. The closest they came to "fast, cheap, and out of control" was with Amundsen and Scott — a pair of ground penetrators that rode along with the Mars Polar Lander in 1999. Ironically, that mission failed, the probes were singled out for being 'too risky.' (If the space agency wants to avoid risk, perhaps they're in the wrong business.) Just thought I should point that out.

Mike Neary

P.S. If you should ever find the DVD, don't fast forward through the parts that don't feature Rodney Brooks, or you'll miss some rather interesting insights.

Thanks for the excellent milestone reminder, Mike.

Bryan Bergeron

Published Monthly By
T & L Publications, Inc.
430 Princeland Ct., Corona, CA 92879-1300
(951) 371-8497
FAX **(951) 371-3052**
Webstore Only **1-800-783-4624**
www.servomagazine.com

Subscriptions
Toll Free **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277, N. Hollywood, CA 91615

PUBLISHER
Larry Lemieux
publisher@servomagazine.com

**ASSOCIATE PUBLISHER/
VP OF SALES/MARKETING**
Robin Lemieux
display@servomagazine.com

EDITOR
Bryan Bergeron
techedit-servo@yahoo.com

CONTRIBUTING EDITORS

Jeff Eckert	Jenn Eckert
Tom Carroll	David Geer
Dennis Clark	R. Steven Rainwater
Fred Eady	Kevin Berry
Mike Keesling	A.J. Neal
Sean O'Connor	Jeremy Kerfs
Bryce Woolley	Evan Woolley
Pete Smith	Fulvio Mastrogiorganni

CIRCULATION DIRECTOR
Tracy Kerley
subscribe@servomagazine.com

**MARKETING COORDINATOR
WEBSTORE**
Brian Kirkpatrick
sales@servomagazine.com

WEB CONTENT
Michael Kaudze
website@servomagazine.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION/GRAPHICS
Shannon Christensen

Copyright 2010 by
T & L Publications, Inc.
All Rights Reserved

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *SERVO Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *SERVO*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *SERVO*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: **430 Princeland Court, Corona, CA 92879.**



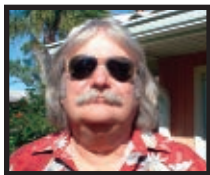
Choose your preferred format:
print or digital media.

We have just what you're looking for!

12 magazine bundle \$57.00
12 issues on CD-ROM \$24.95

Order today 800 783-4624
or online @ www.nutsvolts.com





Bot to Explain Decision Making



UCI cognitive scientist Jeffrey Krichmar (right) and former Hollywood animatronics engineer Brian Cox designed CARL to think and act like a human being. Photo by Daniel A. Anderson/University Communications.

Whereas operational areas of the human pumpkin are fairly well mapped out, "Little is known about the areas of the brain involved in making decisions when faced with uncertainty," observed University of California at San Diego (www.ucsd.edu) cognitive scientist Jeffrey Krichmar. A better understanding could help answer such questions as "Why did I major in fine arts?" "Why did I buy a used Touareg?" "What made me think I could handle eight tequila shooters?"

Enter CARL — a robot with a "biologically plausible nervous system controlled by a realistic model of the human brain," whose mission is to "provide neuron-level insight about the specific brain areas responsible for decision-making and attention, advancing robotic design as well as knowledge of human behavior."

Data for the upcoming \$1.6 million study will be generated by observing the decision-making abilities of rodents that have been confused by screwing around with their food reward stimuli. Brain recordings taken from the rodents will be programmed into CARL's "brain," enabling him to perform rat-like behavior. The assumed link between rodent and human behavior isn't explicitly spelled out, but rumor has it that the American Bar Association is more than a little interested. ♦

Weed Society Considering Robotics

No, we're not talking about NORML or automatic herb harvesting equipment. This is a conservation group called the

Weed Science Society of America (www.wssa.net) which will be holding its annual (okay, pun intended) joint meeting with the Society for Range Management (www.rangelands.org) in Denver, February 8 through 11. The event is expected to draw thousands of scientists, students, educators, and professionals, including Interior Secretary Ken Salazar. More than two dozen symposia are planned, including sessions on how robotic systems and global positioning technology can be used to detect, identify, map, and target weed infestations.

Other topics range from herbicide-tolerant crops and spray drift minimization to the increasing problem of poisonous larkspur growing across the western United States. It may sound a little on the tree-hugging side, but pre-meeting tours include a ski trip, technical tours for hands-on learning experiences, and visits to ranches and rangelands. Details are available at www.wssa.net and www.rangelands.org/denver2010. ♦

From Eyes to Skies

While at Caltech, Prof. Wolfgang Fink helped create Cyclops — a robot that can simulate the experience of a blind person wearing a retinal prosthesis that provides visual stimuli. Cyclops has been highly useful in the development of such devices, because there are so few actual patients to study. For example, the Argus II system from Second Sight (www.2-sight.com) had been implanted in only 18 subjects as of last report.

According to Fink, "We can use Cyclops in lieu of a blind person. We can equip it with a camera just like what a blind person would have with a retinal prosthesis,



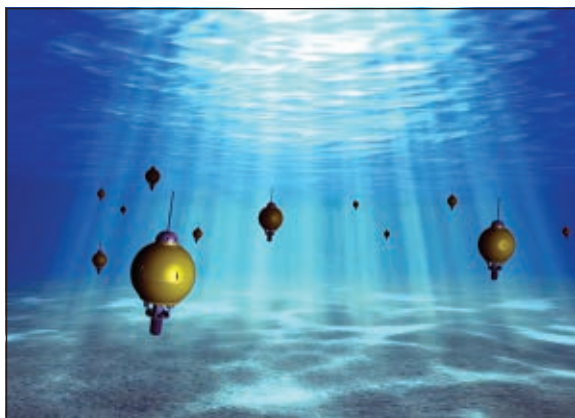
Cyclops is designed to simulate the experience of a blind person fitted with a visual prosthesis.



Poisonous larkspur, a threat to unwitting, hungry livestock. Photo by Jack Dykinga, courtesy of USDA.

and that puts us in the unique position of being able to dictate what the robot receives as visual input.”

But the professor recently jumped ship to join the University of Arizona’s (www.arizona.edu) College of Engineering, where he will head up the Visual and Autonomous Exploration Systems Research Laboratory and adapt Cyclops for a completely different mission: autonomous planetary exploration. Fink says it’s only a short leap, both in terms of hardware and software. All that’s needed is some image-processing system enhancements and a bit of reengineering to make the platform self-maneuvering. Working with other scientists and engineers in UA’s Lunar and Planetary Lab, he hopes to have it ready for desert testing in due course. ♦



Autonomous underwater explorers (AUEs) will provide new information about the oceans.
Courtesy of SIO.

AUEs to Explore Ocean Processes

As reported in these pages from time to time, the Earth’s oceans are already swarming with autonomous underwater vehicles (AUVs). It seems that — although most of them are pretty useful for detailing large-scale ocean processes — a need has been identified to poke around in relatively minute processes. The National Science Foundation has therefore awarded nearly \$1 million to the Scripps Institution of Oceanography (sio.ucsd.edu), in La Jolla, CA, to develop a new breed of vehicle dubbed autonomous underwater explorers (AUEs).

According to the institution, “By defining localized currents, temperature, salinity, pressure, and biological properties, AUEs will offer new and valuable information about a range of ocean

phenomena.” In practical terms, they will “aid in obtaining information needed for developing marine protected areas, determining critical nursery habitats for fish and other animals, tracking harmful algae blooms, and monitoring oil spills.”

The implementation will take the form of systems in which several soccer-ball-sized main units are deployed, each watching over tens or even hundreds of pint-sized units that move in swarms. (Another \$1.5 million has been allocated for developing systems to control their movement.) This is great news for abalone and plankton, but the AUEs look a bit like tasty little jellyfish, so let’s hope larger seagoing creatures can tell the difference. ♦

Saving Bessie’s Butt

The results are in, and Dr. Sarah Baillie, of the Royal Veterinary College, University of London (www.rvc.ac.uk), has been awarded the (London) Times Higher Education’s Most Innovative Teacher of the Year 2009 for her work with haptic simulators. Specifically cited is her “haptic cow,” described as a robotic device used to train students to conduct internal examinations of bovine reproductive tracts.

Apparently, Dr. Baillie observed that it’s difficult to use live animals in teaching because, when a student’s arm is inserted into a real cow’s rectum, you can’t really see what’s going on inside. Her solution is a fiberglass model that contains simulated organs and allows the examination process to be displayed on a computer monitor. Reportedly, the fake cow even “moos” if the student applies excessive force on its private parts.

The doc has also invented a haptic horse for training students in the treatment of equine colic and other abdominal complaints. It was further noted that the concept promotes animal welfare by ensuring that university teaching herds “are not used too heavily or manhandled by novices.” ♦



The award-winning haptic cow.
Courtesy of Royal Veterinary College.

Hit and Run Alarm Bot

Probably the most annoying robotic gadget to emerge this month is Mr. Wake, designed to jolt you out of sweet slumber and then run away before you can clobber him with a shoe. Fortunately, Wake isn't commercially available and, in fact, is still in the development stage as of this writing, being an entry in a letsmakerobots.com contest. Its inventor, Vadim Ryazanov, a resident of Baku, Azerbaijan, says you can build your own by investing about \$80 and 16 hours of labor.

Details are sketchy, but the 900g bot employs two geared motors under the control of a PICAXE-18X microcontroller and is driven by four AA cells. Updated info and videos should be available on the website by the time you read this. **SV**

Mr. Wake, an alarm bot of Azerbaijanian extraction.



Ardweeny. You'd think that's what all the big & mean Arduinos would call the smallest one. ...and you'd be right.

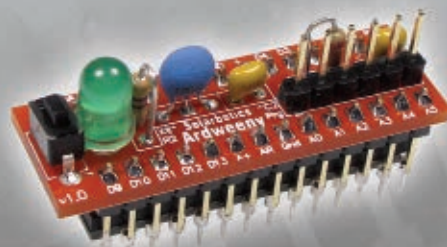
It's the smallest Arduino you can build yourself!

Mount all the parts on the Backpack PCB, and it takes up no more space on a breadboard than the included ATmega328 chip itself!

- Inexpensive • Ideal for Breadboarding
- Fully Arduino compatible • Simple to build!

SOLARBOTICS
www.solarbotics.com 1-866-276-2687

PS- But don't worry, he's like that small kid that gets picked on, but when the bully tries to push him, KAPOW! KARATE-CHOP ACTION TAKES OFF HIS JACK-USB PORT! TAKE THAT, SUKKA!



SKU: ARDW
Price: \$9.95

This image is to scale (...no, seriously!)



Write and Grow Rich

Have you thought of becoming a technical writer? With your technical background, it's an easy way to add an extra income stream. You can even do it full time and create a small fortune.

Top level tech writers easily make over \$40 per hour. And according to the U.S. Dept. of Labor, demand for technical writers is set to explode in the coming years. This could be your chance to create your dream lifestyle! Plus you'll interact with clients who will appreciate your technical nature.

There is one program that gives you a step by step plan to become a wealthy tech writer in the shortest time frame. The best part - you can get started immediately! For more information, go to:

www.fasttracktrain.com/roboticswriter

enter code SR185 for a special discount

"THE PHANTOM DRAW"



The KILL A WATT
meter is the
best way to
help you
determine your
actual energy
draw in ON and OFF
home appliances.

To order call 1 800 783-4624
or online www.nutsvolts.com
\$29.95 plus S&H



Artificial Intelligence Brings Humanoid Robots to Life

Researchers have a variety of goals for artificial intelligence. Some scientists are examining human biology together with artificial intelligence in an effort to create robots that are as humanlike as possible. Still others examine AI from an engineering perspective, begging the question, **what can robots be enabled to do?**



Nao robot shown with human hand (for scale).

The mere building blocks for advancing AI for humanoid robots are many and highly technical. Robotic vision via web cams and machine vision, natural language processing and machine learning, common sense reasoning and knowledge, robotic agility and motion, and robot-to-human-to-robot interaction are a few of the areas of research that will move us closer to more human-like robots. What research areas will have to emerge to take us there? Will we know before we get there?

Let there be light! Robot machine vision leaps forward thanks to real time image processing.

Ten years ago, processor speed was not fast enough to enable real time video image processing. Today, it is. "Real time processing of video images means we can now put cameras on robots and have them understand the world through video images," says Professor Peter Stone, Department of Computer Sciences, University of Texas at Austin.

A decade or more ago, roboticists could only use sonar range finders in a single dimension to discover how far the robot was from the nearest wall. This gave the robots no information about faces for face recognition, no information about

people's location relative to the robot, and no information about color.

Robots moving from legacy sensors to real time machine vision is like asking people to walk around a room with their eyes closed, illustrates Professor Stone. People would likely use their hands as bump sensors to sense objects and feel their way around the room.

This would make moving around slow, complex, and herky-jerky. Humans can do a lot more if they are allowed to open their eyes and process vision in real time. So can robots.

Robot machine vision has made a lot of progress, though not all the problems surrounding it have been resolved. At the same time, there has been progress accomplishing vision tasks using a new three-dimensional laser range finder that retrieves more than 1M range points per second. This gives the robot 3D range pictures of the world around it, according to Professor Stone. "We are researching this here," he comments.

Velodyne is making this range finder — the LIDAR HDL-64E. The 64-element sensor offers 360-degrees of HFOV (horizontal field of view) and 26.8 degrees of VFOV (vertical field of view), according to Velodyne. It reads 1.3M data points per second.

This provides the distance and intensity data about the environment that are necessary to achieve a 3D view of the world.

While the HDL-64E is too big for a small humanoid robot, it might fit onto a full-sized humanoid robot, according to Professor Stone. "We have an HDL-64E on one of our autonomous cars. We are able to get a much richer stream of information than was ever possible before," he continues.

Learn the Language

For humanoid robots to become as intelligent as people, they will have to be imbued with the ability to understand natural human language. This is called natural language processing. "Right now, if you type a question into Google search, it won't simply answer your question. It doesn't understand language in the way that people do," explains Professor Stone.

With robots, the idea would be that they could hear or read natural human language, understand it, and respond with learning, communication, or obedience to a command. With computers, people could stop using keyboards and simply talk to their computers. The computers would be able to listen, parse the



Here's lookin' at you, Nao.

words and sentences, and understand and respond by typing out words in a word processor or initiating computer commands.

"You could just tell your computer, open my browser and search for directions to the nearest restaurant, rather than having to go and do what we do in Google maps," says Professor Stone.

Machine Learning

When robots learn by doing, they no longer need to be instructed or controlled quite so much by an outside intelligence (man's). "With machine learning — which is one of my areas of expertise — the goal is to write and develop algorithms that

News: Humanoids Beat Humans in 2050

Professor Peter Stone of the Department of Computer Sciences, the University of Texas at Austin, is dedicated to the goal of the RoboCup Federation, which is to build a humanoid robot team that can defeat the best human world cup team on a real soccer field by 2050. According to Dan Burrus, founder of Burrus Research Associates, Inc., and a long-time roboticist, it is more likely that Professor Stone will reach his goal by the 2030 to 2040 time frame.

Accomplishing Professor Stone's goal will solve many important challenges and problems around AI for humanoid robotics. Examples include the robot agility problem (having robots run quickly and manipulate their motors in a very granular way so that they can kick a soccer ball), planning, teamwork, and higher level cognitive functioning. "The robots need to be able to reason about their opponents and not fall for the same trick over and over," Professor Stone explains.



Front view of a Nao robot, like the ones Professor Stone and others are experimenting with for RoboCup soccer.

allow computers and robots to improve their performance over time without being explicitly told how to accomplish their tasks," says Professor Stone. Rather than Professor Stone and his students programming their robots to walk faster, to move their legs a certain way, to move into certain joint angles, they programmed in a machine learning algorithm that gave them the ability to experiment with walking; to try different leg motions and joint angles. With this, the robots can record the movements that enable them to walk faster. They can then experiment with combinations of these movements, to see which of those help them to walk faster. In this way, the robots can improve the quality of their behavior in relation to the desired goal.

Processors, Sensors, and Bandwidth

The more powerful the building blocks — such as processing power and speed, sensor capability, and bandwidth — the more scientists are able to think in new and different ways about solving problems about AI. Here, Professor Stone draws an analogy with man's attempt to use computers to play chess:

"When computers were first developed, the notion of a computer that had the speed of Deep Blue really wasn't imaginable. But, at the time, people were trying to figure out whether computers could be made to play chess using the types of processing speeds that people had back then, in the 1970s, which is just orders and orders of magnitude different from what is available today."

"Once you get to the point of being able to do the kinds of search operations that computers can today, then a more or less brute force type of approach, searching billions of chess board positions every second, is made possible, and it opened up completely different ways of thinking about the problem," says Professor Stone.

AI Today

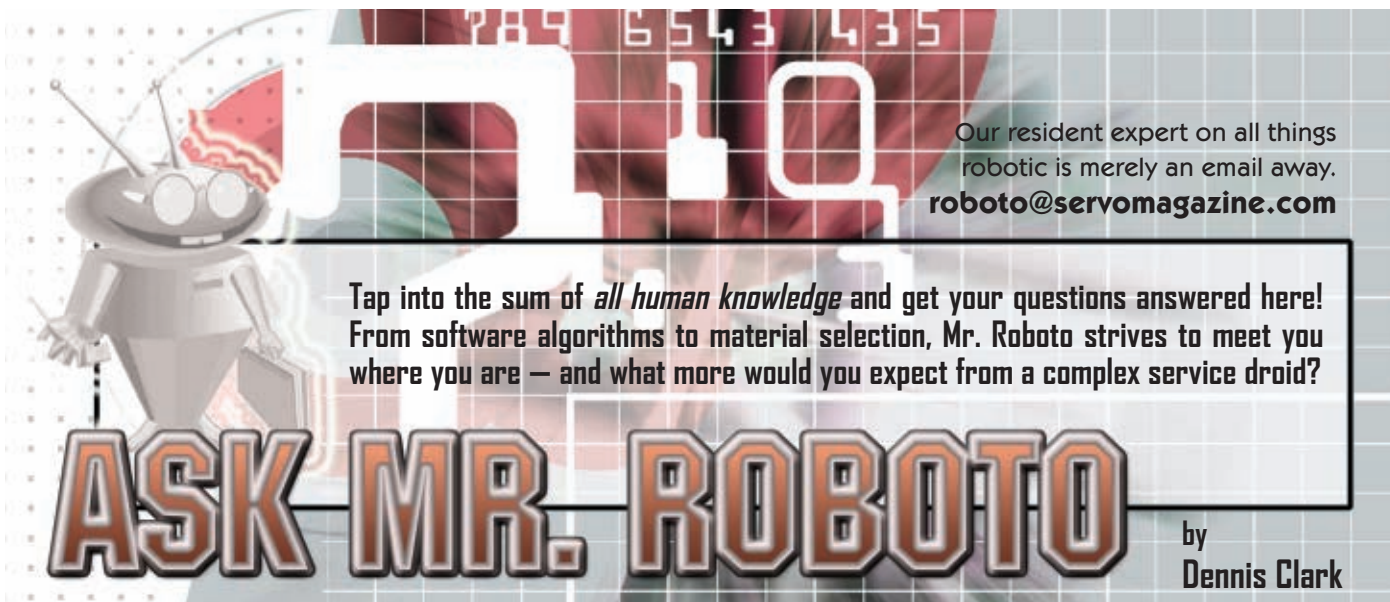
Today, artificial intelligence is good at all the things that modern day computers are good at, such as searching through databases of information looking for the most relevant records. "Humanoid robots really are computers," says Professor Stone. Robots are not better than people at physical motion, though that is what we primarily think of robots as doing.

Robots do not yet have the same degrees of freedom that humans have. Researchers are working on this, including robots with degrees of freedom in their backs, through use of spines that mimic the capabilities of human spines, according to Professor Stone. "But, giving robots spines is difficult and expensive."

Conclusion

In order for truly humanlike humanoid robots to model the capabilities of people, they must be instilled with intelligence for a number of individual tasks and accomplishments. Taken together, however, these different types of intelligence will form building blocks for more fully humanlike behaviors and thinking. **SV**

Video of Nao robots used in RoboCup
www.cs.utexas.edu/~AustinVilla/?p=nao



It is a new year. Part of me shudders (another year gone by) and part of me rejoices (what will be new this year?). Nothing seems to come to a person without some kind of conflict! For me, new stuff in robotics is ALWAYS interesting, but it also comes with a conflict: What about the stuff that I am already used to, know, love, and am good with?

This month's column started out much like that little internal question I pondered above. In our robot group's discussions a little while ago, that very question came up (YahooGroup: FrontRangeRobotics). The group as a whole was discussing how nice the Arduino environment is for the newcomer to embedded programming and amateur robotics. A regular poster (Oric_Dan to those who see his posts all over the 'net) asked if the Arduino could handle software serial ports in multiple numbers. He wanted to know if it could possibly be a step up from that pioneer in the embedded hobbyist world: the Parallax BASIC Stamp. Well, no one knew. A lot of us have used a BASIC Stamp as a central clearing house for our digital I/O from serial inputs over the years, but it's been limited in how much it could buffer. There just wasn't that much RAM in the device. So, I did some investigating, and here are the results.

Q . Can the Arduino open and use several serial ports at the same time so that many different devices can be communicated with from a single microcontroller?

— Joe
Boulder, CO

A . Good question! I've been a fan of the Arduino environment for about a year now. It runs on just about every OS there is out

there, it is open source, has inexpensive hardware, and it is easy to use. What's not to like?

Well, for one thing, the standard Arduino library can only deal with the hardware UART for serial communications. The Arduino has an expansion library called *SoftwareSerial* which adds a single software serial port; but it's only one port and it has some limitations. Fortunately, the Arduino has an incredible open source support network, and a group of folks developed the *NewSoftSerial* library that helps us out. *NewSoftSerial* (NSS) can add any number of software serial ports to your Arduino project — subject to your memory availability, obviously. There are some caveats, however. I quote from the *NewSoftSerial* developer's site: "NewSoftSerial is written on the principle that you can have as many devices connected as resource constraints allow, as long as you only use one of them at a time."

Hmm, what does that mean, "... use one of them at a time ...?" One way of looking at it is that only one block of code can be used at a time so, of course, only one NSS can be used at a time, right? Wrong. Or rather, it depends. I did some experiments to see what would and would not work. I started out small, so let's first see how to add the expansion library to the Arduino environment.

Install the Arduino Environment

Get the latest version for Arduino which is version 0017 from <http://arduino.cc/en/Main/Software> select the one that matches your OS. I am a Macintosh user, so for me I get a .dmg file and just drag the Arduino.app file to a folder

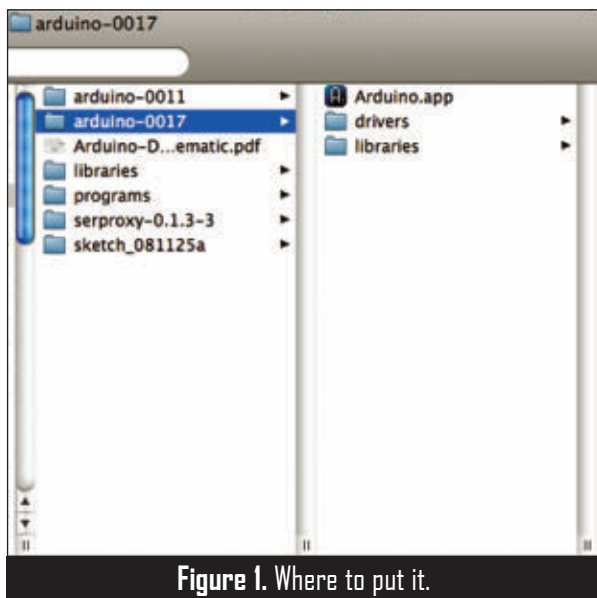


Figure 1. Where to put it.

created under Applications (**Figure 1**).

This location is important since there is a folder under my Arduino folder called *libraries*. Go ahead and create this folder as we'll need it for the next step. I created a *drivers* folder for the two virtual COM port drivers that came with the install. You need to know where your Arduino install thinks it is installed. To do this, start Arduino (double-click on it) and look at your *preferences*. On the Mac, this is under the *Arduino* menu. When you bring up your preferences, it should look something like **Figure 2**. *Sketchbook location* is the folder under which you need to create your *libraries* folder. I didn't change anything in my preferences; I just used what it was set to.

Find and Install the NewSoftSerial Library

Get the NSS library from <http://arduiniiana>.

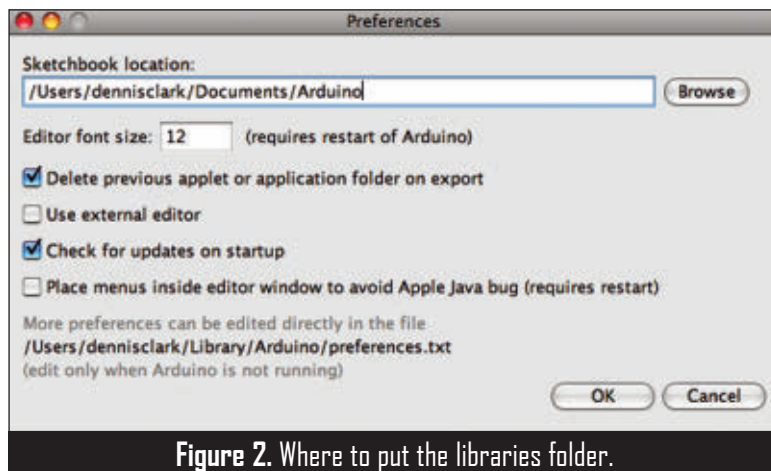


Figure 2. Where to put the libraries folder.

org/NewSoftSerial/NewSoftSerial9.zip.

When you unzip this file, there will be a folder called *NewSoftSerial*. Drag that folder into the *libraries* folder you created previously.

You are now ready to use your Arduino and its new expansion library *NewSoftSerial*. I'm not going to describe how to set up your Arduino hardware. The Arduino site does that very well and it is very easy. I am using my own ATMEGA168 board with an Arduino bootloader variant on it for my tests.

Using the NewSoftSerial Library

When you install the NSS library in the location specified above, you will need to re-start your Arduino application so that it can find it. To use this library, all you need to do is import it. To import a library, just start a new sketch (what Arduino calls a program). Take a look at **Figure 3**. Notice that your file just got a new "#include" entry. That is all there is to it. Now we can start programming with NSS.

There isn't any user documentation that I've found for NSS, so what I know about the library is what I've seen in the two sample programs and from reading the source. NSS appears to only support 8N1 serial and all of the common baud rates from 50 to 115,200. You can define the Rx,Tx lines when you define the NSS object or use the `setRx()` and `setTx()` methods later on. NSS inherits from the standard Arduino `Print` class, so you can print data using the methods defined in the

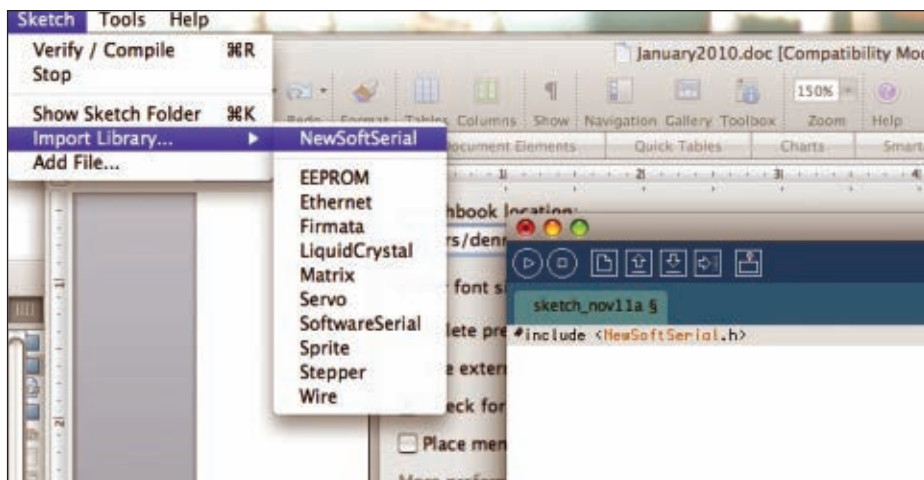


Figure 3. Importing a library.

standard Arduino serial routines, such as print, println, available, begin, read, write, and flush. All of these are well defined in the Arduino references. This is actually very convenient because you can use the same syntax with NSS as you do with the hardware UART-based serial object.

Arduino Programming and NewSoftServo

Whenever I start with a new IDE, compiler, board, whatever, I typically begin by doing something simple. I have a fair collection of simple devices that are configured and used via a serial port. On my first pass, I chose a PWM controller chip that I designed a few years ago. It simply takes a single byte of control information at 9600 baud 8N1 (eight bits, no parity, and one stop bit.) **Listing 1** shows an Arduino program to write commands to this chip. I attached an LED to the output of the chip's PWM so that I could see the LED grow brighter as the PWM value increased. When you define an NSS object, you need to give it a receive I/O pin, as well as a transmit I/O pin. The statement *NewSoftSerial mySerial(2,3);* tells the compiler to configure Arduino I/O pin 2 (PORT D PIN 2) as the serial receive line and Arduino I/O pin 3 (PORT D PIN 3) as the serial output pin. The statement *mySerial.begin(9600);* tells the compiler to configure the serial port at 9600 baud. Write this code into a sketch pad and save it. Then, click on the *compile* and *download* button (**Figure 4**) to download it to your Arduino when your board is in bootloader mode (this depends on your board type and setup.)

Resources

Arduino
www.arduino.cc/

NewSoftSerial Expansion Library
<http://arduiniiana.org/libraries/newsoftserial/>

LISTING 1. Really simple example.

```
#include <NewSoftSerial.h>

/*
 * This is the global variable declaration space.
 * With small memory embedded devices it typically
 * saves on RAM usage to make commonly accessed variable global.
 */
NewSoftSerial mySerial(2,3);           //(receive pin, transmit pin)
int loopер;

void setup()
/*
 * setup() is the first function that the Arduino "OS" calls when
 * a program starts.
 * This is where you put all of your initialization routines and
 * anything that you want to happen only once upon startup.
 */
{
    // set the data rate for the NewSoftSerial port
    mySerial.begin(9600);

    loopер = 0;

    delay(5000);                       //allow hardware to get set up.
}

void loop()
/*
 * loop() is the main execution loop in an Arduino program.
 * This loop simply runs forever.
 */
{
    loopер += 1;
    if (loopер > 120)                   // PWM chip takes values from 0-127
        loopер = 0;

    mySerial.print((char)loopер);      // Here we send the data to the chip
    delay(50);
}
```

```
#include <NewSoftSerial.h>

/*
 * This is the global variable declaration space.
 * With small memory embedded devices it typically saves
 * on RAM usage to make commonly accessed variable global.
 */
NewSoftSerial mySerial(2,3);           // Computer terminal port
NewSoftSerial Display(10,8);           // My serial LCD
char c;

void setup()
/*
 * setup() is the first function that the Arduino "OS" calls
 * when a program starts. This is where you put all of your
 * initialization routines and anything that you
 * want to happen only once upon startup.
 */
{
    // set the data rate for the NewSoftSerial port
    mySerial.begin(9600);
    Display.begin(9600);

    Display.println("hello");

    delay(5000);                       //allow hardware to set up.
}

void loop()
{
    if (mySerial.available())           // if a char is here
    {
        c = mySerial.read();
        Display.print((char)c);        // echo char to serial LCD
    }
}
```

LISTING 2. Serial input and output.

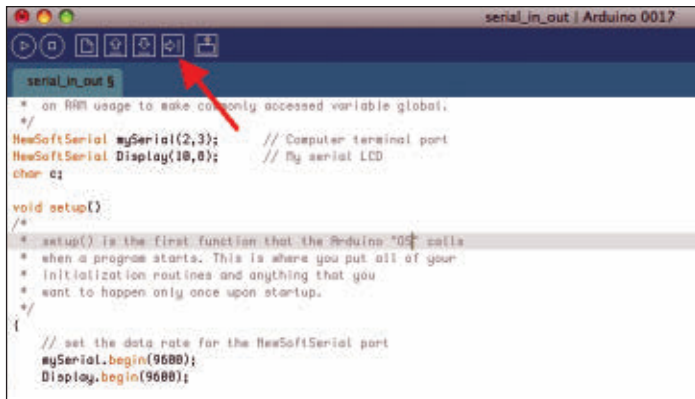


Figure 4. Downloading code.

Pretty easy to program, isn't it? The joy of the Arduino is that it hides the nitpicky little setup details that it takes to get hardware and software to work together. Let's try some code that does serial input, as well as output. **Listing 2** shows how to set up a program that will listen to one pin for serial input and then echo that data out another serial output pin.

The NSS "Gotcha" is Now Discovered

We've done NSS programs using one serial output and one serial input objects. How about multiple NSS objects doing several different things?

Listing 3 shows a program that uses three NSS ports doing output and one doing serial input. I'm using one of my four port serial servo controller chips here to control a servo. This chip has a "sync" pin that you can look at and know that you won't glitch the servo when you write a new value to it. This sync pin is used to pace the whole loop to repeat every 20 ms, more or less. It was here that I discovered the limitations of NSS.

Recall the caveat, "NewSoftSerial is written on the principle that you can have as many devices connected as resource constraints allow, as long as you only use one of them at a time." I just found out what that meant. In this program, it looks like everything is being used one at a time, right? Wrong! NSS is interrupt based which means that it keeps buffers behind-the-scenes and uses interrupts to send and receive data. That means this is happening while other sections of code are running, which means that the poor serial input object listening to its input line is not getting enough time to work well (that's my guess, anyway). All of the serial output routines worked just fine together, but that serial input port is more complex and time-dependent than it appears.

If you want to use both serial input and serial output objects in your NSS-based program, you'll want to be sure that you are using the serial input object when nothing else is active, and that you are done using it before you start banging

```

/*
 * Testing multiple software serial ports and other cool things.
 */

#include <NewSoftSerial.h>

NewSoftSerial Display(10,8);
NewSoftSerial mySerial(2,3);
NewSoftSerial myServo(4,7);

int looper;
int sLoop;
char c;
int syncPin = 9;

void setup()
{
    // set the data rate for the NewSoftSerial port
    myServo.begin(9600);
    mySerial.begin(9600);
    Display.begin(9600);
    Display.println("hello");
    pinMode(syncPin, INPUT);

    looper = 0;
    sLoop = 20;
    delay(5000); //allow hardware to get set up.
}

void loop() // run over and over again
{
    looper+= 1;
    if (looper > 120)
        looper = 0;

    sLoop+= 1;
    if (sLoop > 250)
        sLoop = 20;

    if (mySerial.available())
    {
        c = mySerial.read();
        Display.print((char)c);
        mySerial.print((char)c);
    }

    mySerial.print((char)looper);
    myServo.print((char)0xFF);

    while(digitalRead(syncPin) == 1); //known good state
    while(digitalRead(syncPin) == 0); //wait for sync
    myServo.print((char)0x00);
    myServo.print((char)sLoop);

    if (sLoop == 200)
        Display.println(looper);
    //delay(50); //The syncPin paces at 20ms
}

```

LISTING 3. Multiple NSS objects.

data out on your NSS-based serial output lines, and vice versa. This seems pretty simple in theory, but could be tricky to implement. I think this library is very nice and allows for a really good way to deal with multiple asynchronous serial devices from a single program in a single device.

That was Interesting

The Arduino is a worthy environment to work with for many reasons and there are a lot of folks out there who agree. I come back to it fairly often, since it is so easy to set up and use for quick projects. My Arduino “playground” for this project is shown in **Figure 5**.

There’s no software files to download this month since the Arduino makes using most embedded hardware functions easy to do. Keep building those robots and keep those questions coming to roboto@servomagazine.com. **SV**

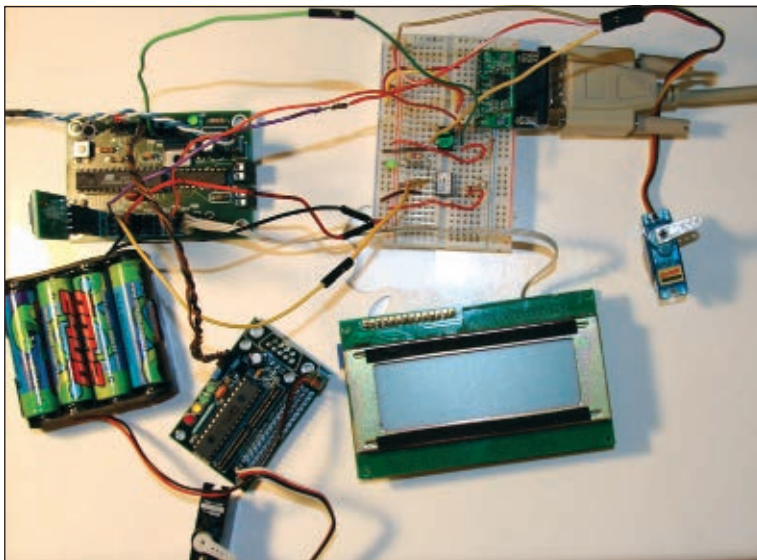



Figure 5. My Arduino playground.



NOW IS THE TIME
To order your subscription to **SERVO Magazine**! Call us TODAY or visit our website to place your order.
877.525.2539 www.servomagazine.com




Combo Price **\$85.55**
SERVO MAGAZINE
Beginner's Guide To Embedded C Programming Book & PIC Kit2 Combo
www.servomagazine.com

NEW! Orangutan SVP-324 Robot Controller

The **Orangutan SVP-324** lets you program your robot in C/C++ with free AVR development software. Use Pololu's AVR C/C++ library to easily make your **motors** go, **servos** turn, **LCD** print, and **buzzer** play — the Orangutan has all of these hardware features (and more) built right in!

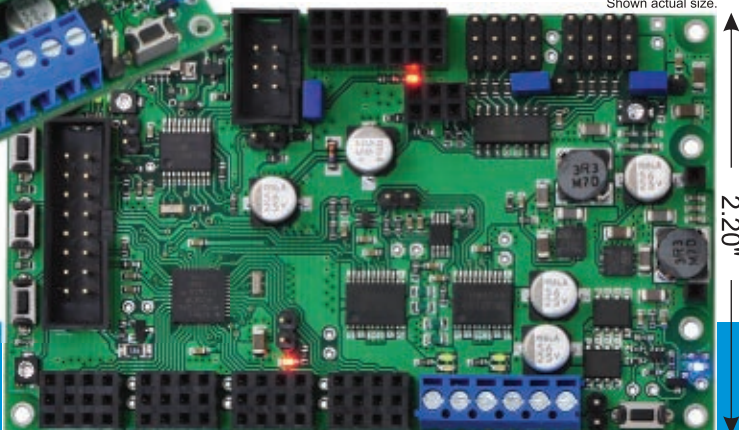
Pololu item #1325 & #1326: \$99.95 and \$89.95

The **Orangutan SVP-324** is available fully-assembled and as a kit.

Key Features:

- * **Wide input voltage range**—6–13.5 V.
- * **C/C++-programmable microcontroller**—User-programmable Atmel ATmega324PA AVR running at 20 MHz (32 KB flash, 2 KB SRAM, 1024 bytes EEPROM).
- * **Built-in USB AVR ISP programmer** (USB A to mini-B cable included).
- * **Two bidirectional motor ports**—2 A continuous/6 A maximum per channel.
- * **8 servo ports**—Use Pololu's AVR C/C++ library to easily control up to 8 servos!
- * **21 free I/O lines**—Up to 12 can be analog inputs, optional dual quadrature encoder inputs, two hardware UARTs.
- * **Removable LCD**—16-character × 2-line, with backlight.
- * **Two step-down voltage regulators**—Each capable of supplying 3 A.
- * **Other fun stuff**—Buzzer, 3 user pushbuttons, 2 user LEDs, and more!

Shown actual size.



Pololu
Robotics & Electronics
3095 E. Patrick Ln. #12, Las Vegas, NV 89120

Save 10%
With coupon code
SVP324SRV3

Learn more at www.pololu.com/svp or by calling 1-877-7-POLOLU.

bots IN BRIEF



VACUUM BITES SNAKE

An article from Israel's *Yediot Acharonot* newspaper, titled “A Vacuum Cleaner Captured a Snake” features a Roomba 560 that appears to have totally obliterated what we’re told was a deadly viper threatening some kids (and possibly a cat) by sucking it up around one of its rotating brushes. There are more graphic pics of the end result over on Facebook, but suffice it to say that the poor little snakey came to a rather violent — and probably really confusing — end. It’s good to know that our robot vacuums have our backs when it comes to poisonous reptiles shaped like electrical cords.



DROP AND GIVE ME 20 ...

Bandit here is a sort of friendly, sort of scary robot designed to help you exercise. (That’s what you want in a personal trainer, right?)

Bandit is helping the University of Southern California Center for Robotics and Embedded Systems conduct a study on exercise training. Seventy volunteers of all ages (including 20 people aged 60 or older living in retirement homes) will have either Bandit himself or Bandit on video as a trainer. The researchers will try to figure out if the physical presence of the robot makes a difference.

FLOATING ON AIR

Remember that Mindflex game from CES last year that used a little ball hovering on a jet of air? Well, if you use a more powerful jet of air and a fancy multi-axis robotic nozzle, you can manipulate objects in three dimensions through obstacle courses, toss things, hold up multiple objects, weigh objects, and even peel onions. It does it all!

The fluid dynamics involved in these manipulations — especially when you throw more than one object and non-spherical objects into the mix — are pretty crazy. Stereo cameras track the object in the airstream and a computer model directs the nozzle to vary the speed and direction of the jet to keep things stable.

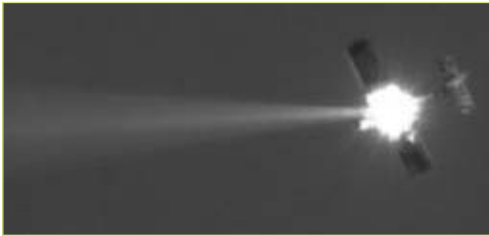
The designers of the system — Aaron Becker and Robert Sandheinrich from the University of Illinois — say this technology could be used to handle flexible and delicate objects in a non-contact setting.



bots IN BRIEF

CREEPY INDUSTRIAL BOTS

Usually, industrial robots look inhuman enough that they aren't scary. Except for ones like these Nextage industrial robots from Japan's Kawada Industries. The robots are mounted on mobile bases, so they aren't restricted to working in just one place. Unlike most industrial robots, the Nextage robots have their sensors and intelligence integrated into one package (complete with stereo cameras in their heads and additional cameras in their hands for examining objects) which makes them more versatile but also much more expensive. These robots are designed to work in concert with humans as part of a cooperative assembly line, but would you want one of these things as a co-worker?



DRONE ZAPPING, LASER STYLE

Boeing's drone-zapping laser cannon has gone live in tests, and has had no problem shooting down not one, not two, but five hapless unmanned drones using a "relatively low laser power" weapon outputting about 2.5 kilowatts. It's not clear how exactly the drones were brought down, though. The deliberately vague press release says that the drones were "acquired, tracked, and negated at significant ranges."

MANTIS FLYS DOWN UNDER

The Mantis UAV from BAE that was at AUVSI in 2009 has taken its first flight out in Australia. The Mantis has a wingspan of 20 meters and is designed to be modular and easy to transport, with capabilities for long range surveillance as well as weapon delivery. Just a few months ago the Mantis was little more than a concept; perhaps deployment will follow the same aggressive schedule.



Cool tidbits and interesting info herein mainly provided by Evan Ackerman at www.botjunkie.com, but also www.robotsnob.com



Zhu Zhu hamsters (L to R) Chunk, Num Nums, PipSqueak, and Mr. Squiggles.

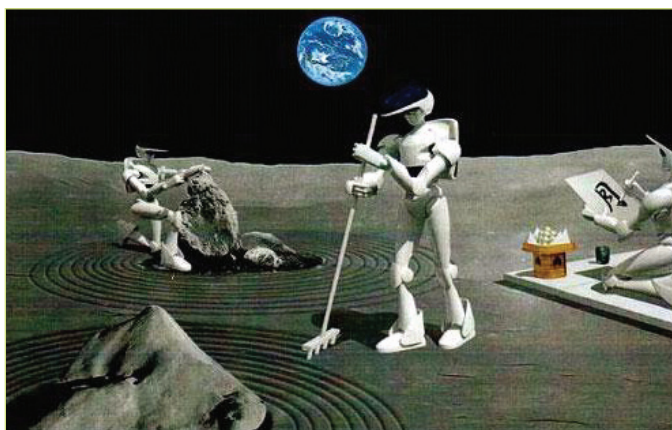
HAMSTERS INVADE

Meet the disgustingly cute Zhu Zhu pet hamster robot, of which there are four varieties: Patches (loves flowers), Chunk (laid back surfer), PipSqueak (petite powerhouse), and Mr. Squiggles (loves to explore). According to vendor Cepia, LLC (www.cepiallc.com), "The best alternatives to real live hamsters, Zhu Zhu Pets hamsters don't poop, die, or stink, but they are still a riot of motion and sound." In addition to running around the house like the real thing, they make 40+ different sounds and, via AI, can vary their sounds to suit whatever room they are in. For example, they make toilet-flushing or tooth-brushing sounds in the bathroom and alarm clock sounds in the bedroom. A dozen accessories are available, including an adventure ball, a funhouse, a hamster mobile with garage, and a surfboard/sleeping hut combo. The little rodents will run you \$8 each separately. Oddly enough, in two- or four-packs, they still cost \$8 each. They were tough to locate last Christmas, being out of stock in many retail and online stores. But they are likely to have multiplied since then and reappeared on shelves. (Now where'd I put that rubber mallet?)

MOON GARDENS

A presentation made by Toyota executives entitled "Realization of Moon Exploration Using Advanced Robots by 2020" discusses sending slick looking humanoid robots to the moon to build rock gardens. Toyota has some fairly specific ideas on hardware and capabilities:

- Joints are protected from regolith.
- Small capacity solar battery onboard.
- Internal status shows on screen on chest.
- Arms exchangeable for different tasks.
- Able to jump with springs in legs.
- Keeps warm during night covered in metal cloak.



There isn't a lot of detail available, but I'd say it's certainly possible to have bots like this in 2020 based on the current capabilities of Toyota's partner robots.

WORLD'S FIRST BIONIC FINGER UNVEILED

Touch Bionics, developer of advanced upper-limb bionic technologies, has announced the commercial launch of ProDigits — the world's first powered bionic finger solution for patients with missing fingers. Now partial-hand patients have a dexterous powered solution to support their return to function and independence.

The ProDigits solution extends life-changing technology to partial-hand patients, who have missing fingers due to either congenital anomalies or amputation from a traumatic incident or medical condition.

Not having fingers or a thumb to act in opposition to one another makes simple tasks such as holding a fork or a cup difficult and frustrating. The articulating digit underpins much of ProDigits' technical advantage and it is this articulation that provides the biggest benefit to the patient. With the ability to bend, touch, pick up and point, the ProDigits reflect the function of a natural hand.

Vince Verges (shown in the photo) was one of four men aboard a Navy EA-6B when it crashed on the Olympic Peninsula on March 19, 1992. He lost all the fingers on his left hand during the ejection.

The nature of each partial-hand patient case is unique, and therefore each prosthetic build is also unique. Sockets are custom-designed and fabricated by clinicians to suit each individual's specific needs.

There are two control strategies that can be employed to power ProDigits: either myoelectric sensors that register muscle signals from the residual finger or palm; or a pressure sensitive switch input in the form of a force sensitive resistor (FSR) or touch pad which relies on the remnant digit or tissue surrounding the metacarpal bone to provide the necessary pressure to activate the finger.



UN BEARABLE

When the St. Louis Zoo lost some of their polar bears, they decided to replace them with robotic ones as part of their holiday exhibit. PETA is dancing with joy, since they don't believe in keeping wild animals in captivity except in natural wildlife reserves to save them. Still, if the trend catches on, zoos may begin to resemble Disneyland's Country Bear Jamboree.

GETTING THE BRUSH OFF

The folks who brought you the Robot Duck have now decided that robots can be useful, as well. Toysmith's Brush Robot, at a size of 6.5 x 9.4 x 2.4", helps you clean up mini messes while ogling you with a curious stare not unlike the money you could be saving if you switched to Geico. The kit comes with wooden and paint brushes, motor, battery case with wires, eyes, and miscellaneous parts.



New XIPMods From Machine Science Make Using an Arduino to Control Networked Peripherals Easier

Provided By Sam Christy
Machine Science

With its low price point, open-source design, and straightforward programming interface, the Arduino has become the platform of choice for many hobbyists, artists, and others looking to get started with microcontrollers. The Arduino board features headers for plugging in simple input and output devices, such as switches, sensors, and light emitting diodes (LEDs). In addition, there are several commercially available Arduino "shields" — printed circuit boards that mount on top of the Arduino headers and make it easy to integrate more sophisticated components, such as DC motors or wireless transceivers. **Figure 1** shows the Arduino Decimila — a common version of the Arduino — which is based on the ATmega168 chip from Atmel.

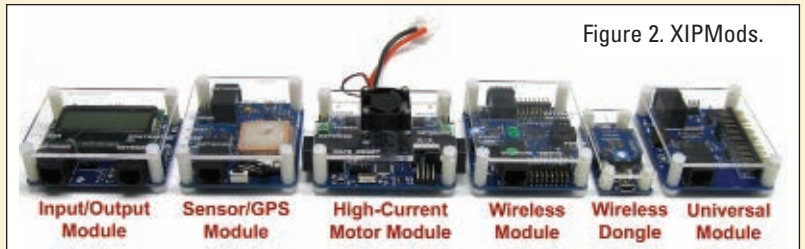
Figure 1.



XIPMods — eXpandable Interlinked Prototyping Modules — provide an entirely new way for users to integrate electronic components with the Arduino. Developed by Machine Science of Cambridge, MA (www.machine-science.org), XIPMods offer Arduino users a suite of inexpensive peripherals, taking advantage of the I²C networking capability of the ATmega168 chip.

Each XIPMod has its own processor, providing efficient multitasking. Multiple modules can be linked together using quick-connect RJ-25 network cables. With the Input/Output Module, (\$79), Arduino users can render text on a liquid crystal display (LCD), control LEDs, gather user input from button switches, produce sounds from a piezo speaker, read signals from an infrared remote control, and track time and date with a real time clock. The High-Current Motor Module (\$129) can drive two 30 amp DC motors, control two 500 milliamp solenoids, and read input from standard hobby R/C transmitters/receivers. The

Figure 2. XIPMods.



Sensor/GPS Module (\$89) adds environmental sensing capability, with built-in light, temperature, and sound sensors, a three-axis accelerometer, and connectors for an optional GPS receiver (\$39) and compass (\$60). Machine Science offers both a Wireless Module (\$89) for XBee-based radio communication, and a Wireless Dongle (\$69) which connects an XBee directly to the user's computer. A Universal Module (\$45) gives users the option to create their own custom device-driver module for components, such as keypads and MP3 players. Code libraries are available for each module, and these can easily be patched into the popular Arduino programming interface. In addition, Machine Science provides free on-line tutorials on the use of the XIPMods (<http://guides.machine-science.org>).

Arduino users can generally set themselves up to access the expandability of this new system within minutes. The instructions in this article are illustrated with the Arduino Decimila, but most versions of the Arduino may be used.

Connecting the Arduino to the XIPMods

An Arduino adapter cable (\$5) links the required power, ground, and input/output pins on the Arduino to the RJ-25 network connector found on the XIPMods. The adapter cable is connected to the Arduino as shown in **Figures 3 and 4**.

Additional XIPMods can be linked together with RJ-25 network cables, which come with the modules. The XIPMods all share a common footprint, so they can be neatly stacked on top of

Machine Science Format	Arduino Format
<pre>#include <machinescience.h> #include <iomod.h> network_control(ENABLE); while(1==1) { iomod_led(1, ON); delay_ms(1000); iomod_led(1, OFF); delay_ms(1000); }</pre>	<pre>#include <machinescience.h> #include <iomod.h> void setup() { network_control(ENABLE); } void loop() { iomod_led(1, ON); delay(1000); iomod_led(1, OFF); delay(1000); }</pre>

one another with threaded nylon standoffs. The XIPMods draw power from the Arduino, which can be supplied either by a USB connection to a PC or by an external power source.

Adding the XIPMod Code Library

To access the XIPMod control functions from within Arduino’s development environment (shown in **Figure 5**), the XIPMod code library must be added.

To do this, simply download the XIPMod code library, shown here. Unzip the folder and copy the unzipped folder to the appropriate directory for your operating system:

OS	Destination Folder for XIPMod Code Library
Windows	../Arduino(version)/hardware/libraries
Mac	../Applications/Arduino(version)/Contents/Resources/Java/hardware/libraries
Linux	../Arduino(version)/hardware/libraries

NOTE: In the Mac OS finder, you may need to right-click on the Arduino folder, and select “Show Package Contents” in order to reach the destination folder.

Installing the XIPMod code library creates a repository of XIPMod code examples within the Arduino IDE. These are available through the Arduino File menu (shown in **Figure 6**), just like any other Arduino code examples.

Writing XIPMod Code for the Arduino

With the XIPMod code library installed, almost all of the XIPMod control functions documented in the Machine Science tutorials are available in the Arduino environment. However, code for the Arduino must be written with a slightly different structure than that presented in the tutorials. With the exception of `#include` statements, `#define` statements, and global variable declarations, all code must be contained in two special functions: `setup` and `loop`. This structure will be familiar to Arduino users.

As an example, consider a simple code for blinking an LED on the Input/Output Module. The version on the left shows how the code would be structured in the XIPMods tutorial; the version on the right shows the structure required for the Arduino.

Both examples contain identical headers: `#include machinescience.h`, which includes code required for all XIPMod projects; and `#include iomod.h`, which includes code required for controlling the I/O Module. The `network_control(ENABLE)` function — which initializes network communication among the XIPMods — needs to be executed only once, so it appears inside the Arduino `setup` function. The code for blinking the LED needs to be executed over and over. It is placed inside an infinite while loop in the example on the left, and inside the `loop` function in the Arduino example on the right. One other minor change is also required: all `delay_ms` functions in the original code file must be changed to `delay` functions for the Arduino. In the example above, the two `delay_ms(1000)` statements become `delay(1000)` statements. **SV**

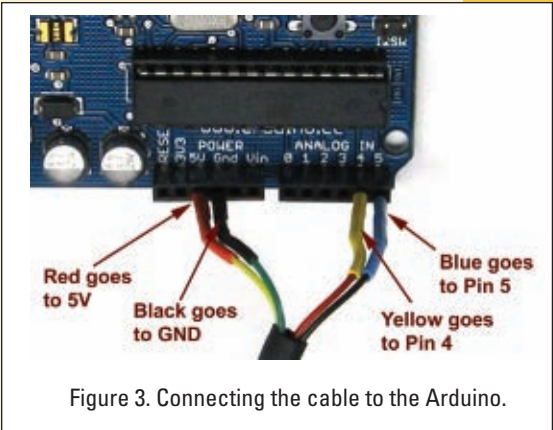


Figure 3. Connecting the cable to the Arduino.

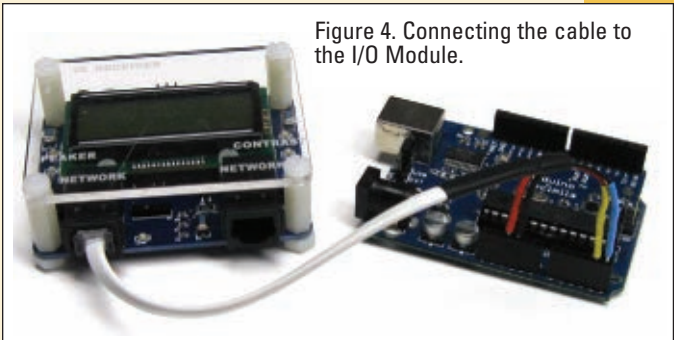


Figure 4. Connecting the cable to the I/O Module.

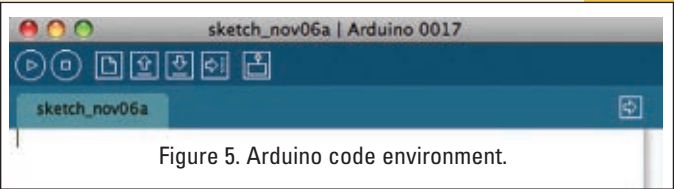


Figure 5. Arduino code environment.

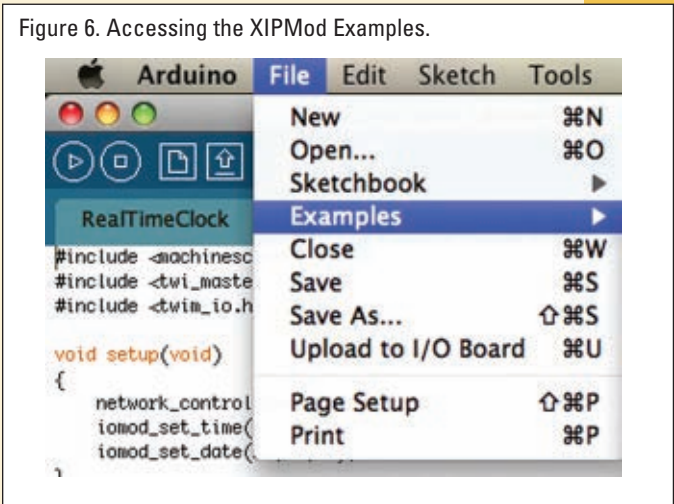


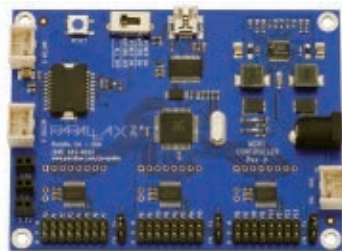
Figure 6. Accessing the XIPMod Examples.

NEW PRODUCTS

CONTROL BOARDS/SENSORS

Propeller Robot Control Board

The Propeller Robot Control Board from Parallax features the multicore P8X32A microcontroller and 64 KB EEPROM, as well as the components necessary to interface with a PC, DC motors, servos, sensors, and more. With the same mounting hole placement as other popular Parallax development boards, it is an easy upgrade for existing projects.



The Propeller Robot Control Board provides all the necessary base circuits needed to build a very powerful mid-size robotics platform. The control board has an on-board USB serial interface to facilitate programming and communication with the Propeller chip. A dual switching supply regulates 6.5 – 20 VDC input to 3.3V and 5V at up to 3A, and contains green and red LEDs to indicate proper operation or an under-voltage condition. The onboard dual H-bridge motor driver makes it possible to directly drive DC motors up to 2.8A and 20V. The 24 available I/O pins are buffered through three eight-bit bidirectional voltage level translators providing direct 5V interface capability. The input voltage can come from a battery pack or a wall adapter using a standard 2.1 mm barrel plug.

The I/O pins are connected to three TXB0108 eight-bit bidirectional voltage level translators. They convert the voltage from 3.3V at the Propeller chip to 5V at the servo headers. These pins are fully bidirectional and are grouped as three ports with eight I/O lines each. Each group is brought out to a set of servo headers. A jumper selects either 5V or VIN for the group. All data pins on the servo header are at 5V signal levels, however should the need arise to directly access the Propeller chip I/O for 3.3V interfacing, solder points are provided to disable the translators and gain direct access to the I/O pins.

The 64 KB EEPROM retains 32 KB for custom data after a Propeller program is loaded. This extra memory could be used for GPS lookup tables, map data, or whatever the user decides. The USB interface provides a serial connection to the Propeller chip through the USB port of your PC. This interface is used for both programming and communication with the control board. Red and green LEDs indicate data flow between the

controller board and the PC. Price is \$99.99.

Sound Impact Sensor



Parallax's new Sound Impact Sensor detects sound from up to three meters away, lending noise activation possibilities to projects. This sensor also includes an onboard potentiometer for easy adjustment of the range of detection for the sensor. This sensor is compatible with all Parallax microcontrollers and sample code for both the BASIC Stamp 2 and the Propeller chip can be found on the Parallax website. Price is \$7.99.

Features:

- Detection range up to three meters away.
- Onboard potentiometer provides an adjustable range of detection.
- Single bit output.
- Three-pin SIP header ready for breadboard or through-hole projects.
- Built-in series resistor for compatibility with the Propeller microcontroller and other 3.3V devices.

For further information on these two items, please contact:

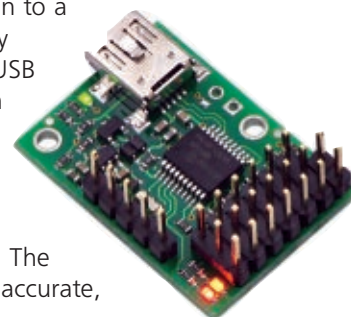
Parallax, Inc.

Website: www.Parallax.com

SERVO CONTROLLERS

Pololu Micro Maestro Six-Channel USB Servo Controller

Pololu announces the release of the Micro Maestro, the first of its new line of USB serial servo controllers. In addition to a TTL serial interface, this tiny board incorporates native USB control for easy connection to a PC and programmability via a simple scripting language for self-contained, host-controller free applications. The Micro Maestro's extremely accurate,



0.25 μ s resolution servo pulses have a jitter of less than 200 ns, making this servo controller well-suited for high precision applications; individual speed and acceleration control for each channel allow for smooth, seamless movements. Units can be daisy-chained with additional Pololu servo and motor controllers on a single serial line.

The Micro Maestro's six channels can be used as general-purpose digital outputs and analog inputs, providing an easy way to read sensors and control peripherals directly from a PC over USB. These analog inputs also enable creation of self-contained animatronic displays that respond to external stimuli.

A free configuration and control program (Windows XP, Vista, and 7 compatible) makes it simple to configure and test the board, create sequences of servo movements for animatronics or walking robots, and write, step through, and run scripts stored in the servo controller. A Linux version of the software is available, and the Maestro is fully Linux compatible.

The unit price is \$24.95 for the fully assembled version (item #1350) and \$23.95 for the partial kit (item #1351).

For further information, please contact:

Pololu Corporation	3095 E. Patrick Ln. #12 Las Vegas, NV 89120 Tel: 877•7•POLOLU or 702•262•6648 Fax: 702•262•6894 Email: www@pololu.com Website: www.pololu.com
---------------------------	---

TOOLS

LabVIEW Education Edition Graphical Software to Inspire Hands-On Learning

National Instruments has introduced LabVIEW Education Edition, a high school focused version of its industry-standard LabVIEW graphical programming software. The new edition is designed to help educators implement project-based, hands-on learning for science, technology, engineering, and math (STEM) subjects. The software was developed in collaboration with the Tufts University Center for Engineering Education and Outreach (CEEEO), a leader in integrating engineering into K-12 education. Working with Dr. Chris Rogers, Tufts CEEEO director and professor of controls and mechatronics, National Instruments developed the software to help high school teachers give their engineering students hands-on experience with the same graphical system design technology used by engineers and scientists throughout industry and academia.

"Engineering is a powerful motivator for learning math and science as it makes it possible for students to be innovative and creative while learning," Dr. Rogers said.

"Computers are at the core of almost every experiment, robot or data analysis, and the graphical nature of LabVIEW facilitates this creativity and innovation on the programming side, even for beginning students. I look forward to the day when the teachers are the skeptics and the students use their math, science, and engineering toolbox to create their own experiments to convince the teacher."

LabVIEW Education Edition helps teachers implement engaging, project-based learning as well as connect theory to real world examples. The new version works seamlessly with key educational hardware platforms such as LEGO® MINDSTORMS® Education NXT, Vernier SensorDAQ, and TETRIX™ — a metal robot-building system by Pitsco — making it easy for teachers to integrate hands-on robotics, measurements, and data acquisition into their curricula. The software's graphical drag-and-drop paradigm helps students learn key programming concepts and develop analytical skills while gaining experience with technology used in the professional world. The new edition also features its own classroom-ready tools, including the Data Viewer which graphically displays sensor data; a virtual oscilloscope; and other virtual instruments that give students hands-on experience with a variety of electrical and mechanical engineering techniques. Additionally, LabVIEW Education Edition includes a supporting curriculum and classroom activities directly available from National Instruments, Vernier Software and Technology, and LEGO Education.

"National Instruments is inspiring the next generation of innovators that will tackle society's critical challenges," said Ray Almgren, vice president of academic marketing at National Instruments. "As a result of our collaboration with Tufts CEEEO, LEGO Education, Vernier, and others, LabVIEW Education Edition gives educators a learning platform to help their students explore robotic designs, measure phenomena, and learn a variety of engineering concepts. By customizing LabVIEW features for classroom use, we believe that this new edition will inspire and engage students and help secure a bright future for STEM education."

Readers can view a webcast about LabVIEW Education Edition features, specifications, and curriculum by visiting www.ni.com/academic.

For further information, please contact:

National Instruments

Website: www.ni.com

Show Us What You've Got!

Is your product innovative, less expensive, more functional, or just plain cool? If you have a new product that you would like us to run in our *New Products* section, please email a short description (300-500 words) and a photo of your product to: newproducts@servomagazine.com

COMBAT ZONE

Featured This Month:

Features

28 PARTS IS PARTS:
My First Welder (or not?)
by Kevin Berry

29 MANUFACTURING:
Debugging Welding Problems
Edited by Kevin Berry

Events

30 Oct/Nov 2009 Results and
Jan/Feb 2010 Upcoming
Events

31 EVENT REPORT:
*No Gain Without Pain —
Franklin Institute 2009*
by Pete Smith

PARTS IS PARTS

My First Welder (or not?)

● by Kevin Berry

Most bot builders, at some time, think about buying a welding system. There are dozens of choices out there, each with their own jargon, culture, and opinions (think Ford vs. Chevy!). Trying to make this easy on newbies (like myself), we asked several experienced welders the following questions:

"If someone needs a 110 volt unit to do basic welding on a hobbyweight bot, what should they buy? What if someone is moving into bigger bots and wants to start with a 220 volt unit? We are talking about new builders, so

best value (as opposed to cheaper) is part of the criteria."

As expected, there were different answers but after some quizzing, a few winners emerged. For the 110 volt class, a used Hobart Handler 135 appears to be a great value. This model isn't made by Hobart any more, but a decent used one can be found on eBay or Craigslist for around



\$300. Of course, like all welding systems, the cost of protective equipment and accessories will run up the cost a bit.

For a new purchase at this level, Hobart's Handler 140, or Lincoln's 140C are similar, and will run about \$700 or so, ready to use. Both seem to be quality products, recognizing that these are for light welding jobs, not tacking together 3/4" steel plate.

For serious bot builders wanting to spend serious money, the Lincoln Invertec V160 stands out. This will run around \$1,400-\$1,600. This 220 volt system is on the lower end of the "big boy" welding scale, but

definitely meets the criteria of "Best Value" for an entry level tool.

So, doing the math on a used Handler 135, adding in gloves, mask, hand tools, some clamps, a welding table, wire, gas ... My allowance is \$20 per week ... carry the seven ... and — well maybe for Christmas next year. **SV**

Lincoln Invertec.



MANUFACTURING: Debugging Welding Problems

● by Some RFL Forum Bunnies, Edited by Kevin Berry

The Problem

The poster has some very old welding experience, from 15 years ago. He just bought a battered old arc welder, 220V AC, with no DC conversion unit. He was using 1/8" 6011 rods on mild steel, and mostly wound up welding the rods to the work. In striking an arc, he was having difficulty keeping it going. He was just trying to practice striking and maintaining a consistent arc, creating a weld pool, and so on.

The Delphi Robot Fighting League forum (forums.delphi.com/THERFL) — a bonanza of fabrication information, opinions, friendly abuse, and a wide (some say wild) range of knowledge — responded enthusiastically with advice.

Here are the nuggets I dug out of their avalanche of useful information:

- Obvious, but it must be said:

Make sure the welder is working properly.

- If you are attaching a lot of rods to your practice piece, it could

Billy Moon's Welding Tips for Newbies

- Metal inert gas welders are the easiest to weld steel with.
- Aluminum and titanium really need Tungsten inert gas welders.
- If you have a joint that will take a lot of flexing (like in a bot or race car), consider brazing rather than welding.
- You must get a digital welding helmet with a variable darkness setting. This makes welding for a newbie way easier. They are reasonably priced these days — a lot of places have "trade up sales" where you can trade in an old, traditional helmet for \$30-50 off a digital one.
- Use direct current to weld upside down (stuff overhead).
- Look up a local welding supply shop in your town and go visit them at lunch time. They will carry good quality welders, welding supplies, and gases, and can offer a lot of advice. You won't get much help from a 'big box' type store.

mean you are feeding the rod too fast for the current/voltage that your machine is putting out.

- It could be a duty cycle thing. If you're getting a good arc for X amount of seconds and it goes away without you changing the way you are welding, it could be the machine taking a break. Some smaller machines have a pretty low duty cycle and are meant more for welding tubing and things like that together where you are going to have several start/stops to give the machine some time to cool down.
- If the rods stick when you start, you probably just need to practice some more. If they stick when you're down the bead a ways, it's a feed rate/current issue.
- There are two ways to strike an arc. The first is to lower the rod (almost touching the plate) and then bring it back up again when it starts to arc. Then, move the bead along (this way is harder but makes a cleaner looking start). The other is to

move the electrode almost like its a match you are trying to light, gently sweeping it to start your arc.

- You want to be running around 100 amps or more for 1/8" 6011. Make sure you are using the right amperage for the stick you have. There are lots of charts available for that.
- Some rods don't like moisture and need to be kept in an airtight box. The 6011 rods do not need to be kept in an oven and, in fact, want about 10% moisture.
- One teacher starts students off with 7024 1/8" rods. It is considered a good rod for learning because you can run it really hot — 150+ amps — and it will almost never stick.
- A practice tip: Using a full stick when first starting out tends to cause an issue with the angle that you begin at. You tend to hold it higher, making it perpendicular to the work, so it sticks. Using the tap method may be best for a repeatable start, while angling the

stick left or right to expose an edge instead of the center of the stick is best.

- Another practice tip: Try using half length sticks for more control.
- Be sure to tamp before you clamp to have clean access to the tip.
- Since this is an old welder, inspect all of the terminals for tight connections and for rust (or oxidation of the copper); the welding cables can develop rust also. The internal connections inside of the welder can rust and even the plug that goes into the wall can develop rusty terminals.
- Clean, clean, clean! Dirty metal doesn't weld well.

Also, I heartily endorse these books:

- *The Welder's Handbook* (It's on Amazon!)
- Download Miller Welds' handbook at: www.millerwelds.com/pdf/guidelines_smaw.pdf. **SV**

EVENTS

Completed and Upcoming Events

Completed Events for Oct 14 to Nov 11, 2009

Gulf Coast Robot Sports-3 was presented by Gulf Coast Robot Sports in Bradenton, FL on October 24th.



Halloween Robot Terror was presented by California



Insect Bots in Gilroy, CA on October 24th.

Mecha-Mayhem 2009 was presented by The Chicago Robotic Combat Association in Rosemont, IL, October 23rd through 25th.



Upcoming Events for January-February

Kilobots XVI will be presented by Saskatoon Combat Robotics Club in Saskatoon, Saskatchewan Canada on January 16th and 17th. Go to www.kilobots.com for more information. **SV**



EVENT REPORT: No Gain Without Pain *Franklin Institute 2009*

● by Pete Smith

The Franklin Institute in Philadelphia, PA (www2.fi.edu) holds their "World Space Week" each autumn and hosted the Northeast Robotics Club's (www.nerc.us) big autumn competition on Saturday, October 3rd.

The Institute is a modern style science museum. It is housed in a magnificent building in downtown Philadelphia. The beautiful rotunda is dominated by a very large statue of a seated Benjamin Franklin (Figure 1). It also boasts an IMAX movie theater, and the main stairwell has a working Foucault's Pendulum (Figure 2).

Robot Day at the museum was not just a combat event. Several FIRST teams demonstrated their bots (Figure 3), Replicas of B-9 from the TV series *Lost in Space*, and multiple R2D2s (Figure 4) were there, along with some props from the *I, Robot* movie, including a brown-eyed Sonny — and lots of other displays including an interesting new angle

on Children's Parties: Robogladators (www.robogladators.com). They had various fighting party bots and a self-balancing (Segway style), two wheeled "Knight" (Figure 5).

Last year, the combat event was run over two days but this year was condensed into just one hectic day of action. Competitors started arriving before 8.00 AM on Saturday morning and fights got going by about 10.30.

There were five different weight classes taking part, from the 1 lb Ants up to the 30 lb Featherweight and Sportsman classes. Teams had traveled from as far away as the Carolinas to take part in what is shaping up to be the premier autumn event for small bots in the East.

The arena and pits were in a different section of the museum than that of the last couple years. The new location made for easier viewing of the arena but meant that the pits were a little tight for space.

The Ants fought round-robin with Drum Poco Tambor dominating; last year's winner Otis took second and Decidedly Undecided took a creditable third.

The Beetle class was also round-robin. Titanium clad drum bot The Rolling Pin (Figure 6) won through to first place by combining reliability

and a little bit of good luck. Blade Spinner Maniac Kathy took second, losing only to the winner when its blade came loose. Rugged wedge Cloud of Suspicion took third place.

A larger field of 12 lb Hobbyweights allowed the usual double elimination competition. Last year's winner Scurrie (Figure 7) and last year's second place (and twice Motorama Champion) Surgical Strike combined with recent arrival from the west coast Fiasco (Figure 8) and also Deranged (with its new "in drum" brushless motor; Figure 9) to form a tough field.

A notable fight was Deranged vs. Flatline where Flatline absorbed a couple of huge hits to win when his adversary ended up inverted. Scurrie repeated last year's reliability to fight its way through tough fights against Fiasco and Surgical Strike to take first place for the second year in a row. Flatline grabbed second and a problem plagued Surgical Strike got third.



FIGURE 1

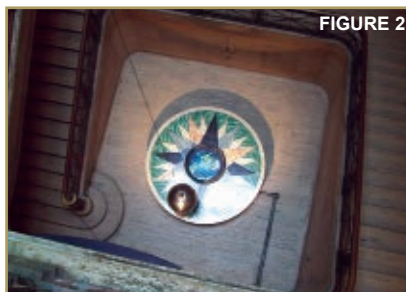


FIGURE 2



FIGURE 3

FIGURE 4



In the Sportsman class, the crowd loved the insect-like Herald but the powerful pneumatic flipper Upheaval dominated to take first, last year's winner Mr. Gadget was second and

FIGURE 5



the hammer bot Mangi was third.

The Featherweights were won by tough wedge Pinball with second place taken by the ever reliable and noisy Gnome Portal; lifter Jagaro

FIGURE 6

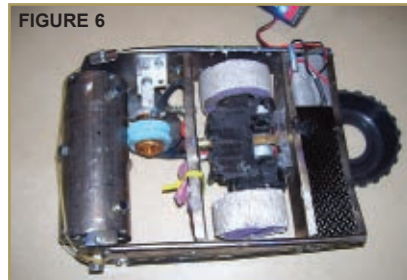


FIGURE 7



FIGURE 8

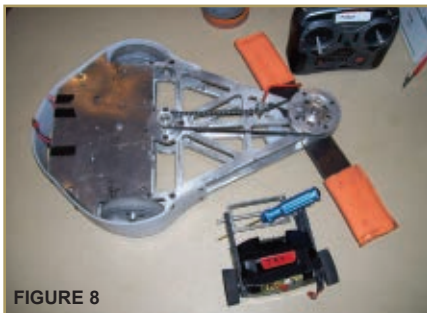
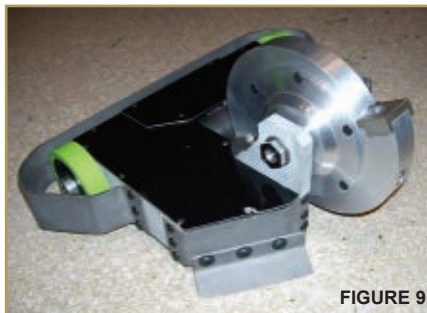


FIGURE 9



took third.

Deranged was awarded a special prize for "Best Hit" and Herald got the coveted Franklin Cup for being judged the most innovative.

The event seemed popular with museum visitors with the seats usually full throughout the day (Figure 10).

All fights were over by 5:30 PM and after a short awards ceremony, folks tidied up their pits and got ready to leave. Already there was talk of the next, much larger event at Motorama in February 2010 and plans for the new and improved bots that would make that competition even better.

For future events, check out www.buildersdb.com the RFL <http://botleague.net/>, and the NERC www.nerc.us. Come and watch, or build a bot and compete! **SV**

FIGURE 10



You can find many of the fights mentioned in the article on Youtube by searching on the bot's name and "Franklin."

**MOTORS
GEARBOXES
WHEELS
AND MORE**

BB BaneBots BANEBOTS.COM 970-461-8880



Tips For Selecting DC Motors For Your Mobile Robot

By AJ Neal

When building a mobile robot, selecting the drive motors is one of the most important decisions you will make. It is a perfect example of an ideal world meeting the real one. This article will cover some of the basic physics and the rules of thumb I use to select DC drive motors for mobile robots. Before you can select your motors, you'll need to know what characteristics the robot you want to build will have. How large will it be? How much will it weigh? How fast will it move? What terrain will it operate on?

Once you have an idea of what your robot will look like, start with some basic physics to get an idea of your motor requirements. After you have calculated your requirements, you can further define the motors that will best suit your robot. Finally, you can start looking for motors that will meet your needs armed with realistic specifications and an understanding of how they relate to your robot's performance and capabilities. In most cases, you will not find the perfect motors for your robot and you will need to make some tradeoffs in order to make your final motor selection.

Let's start at the beginning with some definitions. What exactly is a DC motor? A direct current (DC) motor consists of a set of magnets, rotor coil, and commutator. When you apply current to the rotor coil, it will turn into an electromagnet and repel the magnets. The commutator causes the current in the rotor coil to switch polarity as it

rotates. This polarity switch causes the rotor coil to repel the magnets and generate continuous torque. Speed in a DC motor is proportional to the voltage applied to the rotor. The power produced by the motor is proportional to the voltage multiplied by the current. When working with DC motors, it is essential to remember the relationship between power, voltage, and current.

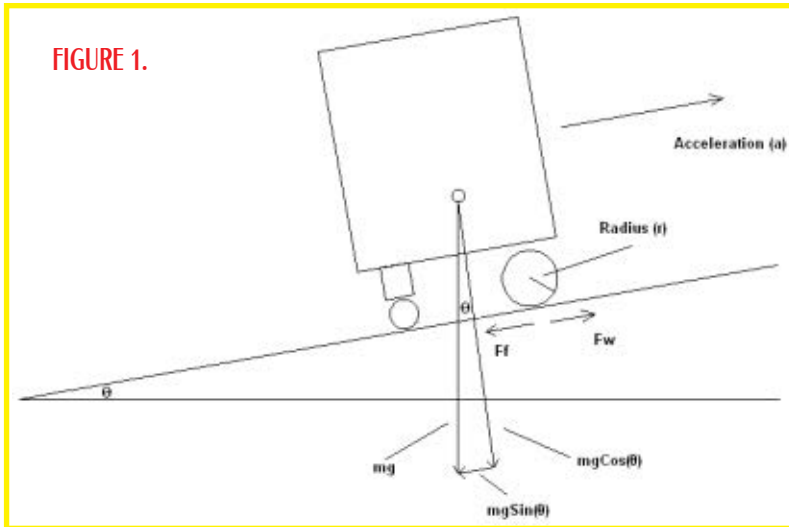
$$\text{Power} = \text{Voltage} \times \text{Current}$$
$$P_{\text{watts}} = V * I$$

Another key relationship is:

$$\text{Power} = \text{Torque} \times \text{Angular Velocity}$$
$$P = T * \omega$$

This means that in order to increase the power output of the motor, you can increase the voltage rating or

FIGURE 1.



you are planning on building a robot to roam around on the Moon or Mars, the acceleration value will be different. This means your robot's weight differs depending on what planet it is roaming around on. Let's take a robot that weights 25 lbs here on Earth. Remember, pounds are the English unit of weight, not mass. Each of the engineering equations we will use later has specific units. It is important to watch out for this. Nothing would be more embarrassing than sending a robot to Mars because weight was used when you should have used mass. **Figure 1** is a simple free body diagram of a simple robot with two front drive wheels and a rear caster. This demonstrates the various forces that will act on the robot. We will start by examining the effect of gravity on our robot. What is force?

$$Force = Mass \times Acceleration$$

Most commonly written as:

$$Force = ma$$

On the free body diagram, the key forces are:

$$Weight = mass \times acceleration = mass \times gravity = mg$$

This is the force pulling the robot down toward the center of the Earth due to gravity.

On a free body diagram, the force of weight is broken down into its two components.

$$Force \text{ Pulling Robot Down Incline} = f_g = mg \sin \theta$$

The force is pulling your robot back down the incline and must be overcome by your drive motors. The greater the angle of incline, the greater this force will be. The incline a robot is trying to climb makes a significant difference in the torque required from your drive motors.

$$Force \text{ Pulling Robot to Incline} = f_n = mg \cos \theta$$

This force is holding your robot onto the incline. This force is required along with friction to allow your robot's drive wheels to push the robot forward up the incline. We will ignore friction, because the force of friction is what the drive wheels need to push to move the robot forward.

On our free body diagram, torque is the force at the edge of the drive wheels pushing our robot up the incline.

But What is Torque?

Torque is the measurement of the force applied to rotate a body around a particular axis. In a mobile robot, the body will be the wheel and the axis will be a motor

increase the current. For example, a 12 volt DC motor can supply the same power as a six volt DC motor, but at 1/2 the current. This is important because most components are limited by the amount of current they can carry. If your robot will be extremely heavy, you may even want to look at 24 volt DC or even 90 volt DC motors. One of the trade offs for the higher voltage is safety. It is hard to shock yourself at 12 or 24 volts, but 90 volts can cause shock and possible injury. Another key property of DC motors is that the speed is controlled by changing the voltage. When sizing a DC motor, we will use the rated voltage of the motor. This is the maximum voltage the motor is designed to handle. There are several different types of DC motors to select from. In most cases, I use brushed DC gearhead motors. Gearhead motors have a gearbox installed as part of the motor. This gearhead is a gearbox attached to the output shaft of the motor. A few other types of DC motors include brushless and stepper motors.

Gearboxes are sometimes called reducers because the output shaft of the gearbox will be less than the output shaft of the motors. The reduction in speed results in an increase in torque. This is a good place to start because most DC motors have output shaft rpms (revolutions per minute) of several thousand and very little torque. Using a gearbox will reduce the shaft speed and increase the torque.

When selecting DC motors, you must understand some of the basic physics that will affect your robot. Some of these physics concepts you should understand are force, weight, mass, torque, acceleration, and velocity, and the relationships between them.

What is Weight?

Most of us have talked about weight all of our lives. We know that every object has a weight. But how is weight defined in the world of physics? Weight is actually defined as the force due to the acceleration of gravity on a body. On Earth, we normally use 9.8 m/(s²) or 32 ft/(s²). If

shaft. Torque is measured in units of force and distance, for example in-lbs. What does this mean? If you have a torque of 6 in-lbs, you can expect 6 lbs of force, one inch from the shaft of the motor. In our case, our drive wheels are six inches in diameter. This means that the force is applied three inches from the drive motor. We could expect 2 lbs of force to be applied by the wheel to the ground. As you will see, many of these values come in a variety of units. Some popular measurements of torque are ft-lb (foot-pounds), oz-in (ounce-inches), and Nm (Newton-meters). Some units will be in English units and some will be in metric units. Values will need to be converted to metric before plugging them into the equations. In most cases, these resulting answers will need to be converted to English units in order to determine what motor to order from most suppliers.

$$\text{Torque} = \text{Force} \times \text{Distance}$$

Finally, velocity is the speed at which our robot will move up the incline and acceleration represents how fast our robot will reach the desired velocity.

How does acceleration relate to speed (velocity)?

$$\text{Velocity} = \frac{\text{Acceleration} \times \text{Time}}{2} + \text{initial velocity}$$

Now that we understand the forces acting on our robot, we can begin the process of sizing the drive motors.

To determine what size motors we need for our robot, we will need to define the following:

Weight of the robot: $w = 25 \text{ lbs}$

Maximum Speed: 60 ft/min $v = 1 \text{ ft/s}$

Maximum incline to climbs: $\theta = 10 \text{ degrees}$

Reach maximum speed in two seconds: $a = .254 \text{ m/s}$

Drive wheels will be six inches in diameter: $r = 3 \text{ in}$

From our free body diagram, we will focus on the forces working in parallel to the inclined surface. Let's also assume that our robot will start from rest and need to accelerate up the incline to full speed.

$f_w = \text{the force pushing against the wheel}$

$f_g = \text{the force pulling robot down incline due to gravity}$

$$\text{Torque} : T = f_w \times r$$

In physics, all forces must balance which gives the equation:

$$\Sigma \text{ Forces} = 0$$

If your robot is moving at a constant speed, the summation of all forces will equal zero.

$$\Sigma \text{ Forces} = f_{\text{total}} = f_w - f_g = 0$$

To properly size our motor, we will focus on the situation where the robot is accelerating from rest to full speed. This is where you want to size your motors to be large enough to get the job done. The torque required to get your robot moving can be much greater than keeping it in motion. In this case, the summation of the forces acting on our robot will equal the total mass multiplied by acceleration. I usually accelerate my robot to full speed in one second or less in the calculations.

$$\Sigma \text{ Forces} = f_{\text{total}} = f_w - f_g = Ma$$

$$f_w = Ma + f_g$$

$$T/r = Ma + M \sin \theta$$

Finally:

$$T = M(a + g \sin \theta)r$$

We must convert weight to mass in metric units:

$$M = 25 \text{ lbs} \times ((1 \text{ kg})/(2.2 \text{ lbs})) = 11.36 \text{ kg}$$

Next, we convert radius from three inches to meters.

$$r = 3 \text{ in} \times \frac{2.54 \text{ cm}}{\text{in}} \times \frac{100 \text{ m}}{\text{cm}} = .0762 \text{ m}$$

$$T = (11.36 \text{ kg}) \left(\frac{.254 \text{ m}}{\text{s}^2} + \left(\frac{9.8 \text{ m}}{\text{s}^2} \right) \sin(10) \right) \times .0762 \text{ m} = 1.69 \text{ Nm}$$

Most of the motors I have worked with define torque in in-lbs or oz-in. We will convert to in-lbs:

$$T_{\text{ft-lbs}} = 1.69 \text{ Nm} \times \left(\frac{.225 \text{ lb}}{1 \text{ N}} \right) \times \frac{100 \text{ cm}}{1 \text{ m}} \times \left(\frac{1 \text{ in}}{2.54 \text{ cm}} \right) = 14.97 \text{ in-lb}$$

This is the total torque required to drive the robot. Since we are using two drive motors, we can divide this in half.

$$T_{\text{per motor}} = 6.5 \text{ in-lb}$$

Next, we will determine how fast in rpms the motor will need to turn.

$$\frac{\text{Rev}}{\text{Min}} = \left(\text{Velocity} \frac{\text{ft}}{\text{min}} \right) \times$$

$$(2 \times \pi \times \text{Drive wheel radius}) \times \left(\frac{1 \text{ ft}}{12 \text{ in}} \right) = 38 \text{ Rev/Min}$$

Finally, to determine how much power the motors are required to supply, the following equation should be used:

$$P = T \times \omega$$

$$\omega = \text{angular velocity}$$

Angular velocity is measured in radians per second.

$$\text{One Revolution} = 2 \times \pi \text{ radians}$$

$$\omega = \frac{38 \text{ Rev}}{\text{Min}} \times \frac{2\pi \text{ Rad}}{1 \text{ Rev}} \times 1 \frac{\text{min}}{60} \text{ sec} = 3.98 \text{ rad/sec}$$

$$P = 1.69 \text{ Nm} \times 3.98 \text{ rad/sec} = 6.72 \text{ watts}$$

If looking at motors with English units, power will be specified in horsepower.

$$P_{hp} = 6.72 \text{ watts} * \frac{1 \text{ hp}}{746 \text{ watts}} = .009 \text{ hp}$$

$$Power_{per \text{ motor}} = .0045 \text{ hp}$$

These values are in a perfect world. Most robots do not get to operate in a perfect world, so you will need to account for the losses in the DC motor and inefficiencies in the gearhead. I usually assume the losses are 50% or greater. Find a motor that will deliver at least twice your requirements or greater; this is a good rule of thumb.

I went to Grainger's website (www.grainger.com) to see if I could find a motor to meet our specifications. The motor specifications usually include the full load current, rpm, and torque. These are the maximum values the motors are designed to operate at. I was able to find one motor that looked like it would meet all of the requirements; the full load torque of 25 in-lbs and rated for 41 rpm. I then noticed it would cost more than \$150 for each motor. Spending \$300 for drive motors would bust my robot budget, so I decided to keep looking. I was able to find two more motors that were close to our calculated requirements: one with an output torque of 10 in-lb at 50 rpm and the other with an output torque of 20 in-lb at 25 rpm. These both used the same motor with a different gearhead and these motors were less than \$50 each. This is a much more reasonable price. Our ideal robot would have at least 16 in-lb of torque per motor at 38 rpm. I decided to go with the motor that would supply 10 in-lb of full load torque at 50 rpm. This would give the robot plenty of speed, but might require the final weight to be reduced. Trading a little weight to save \$200 seemed like a good deal to me. I also know that my specification is for a "worst case" situation and my robot will probably handle a 25 lb robot just fine.

Once you have selected your motors, you can plug them back into your equations and see how your choice

may affect your final robot design. Again we start with the equation:

$$T = M (a + g \sin \theta) r$$

A little more algebra takes us to:

$$M = \frac{T}{(a + \sin \theta) * r}$$

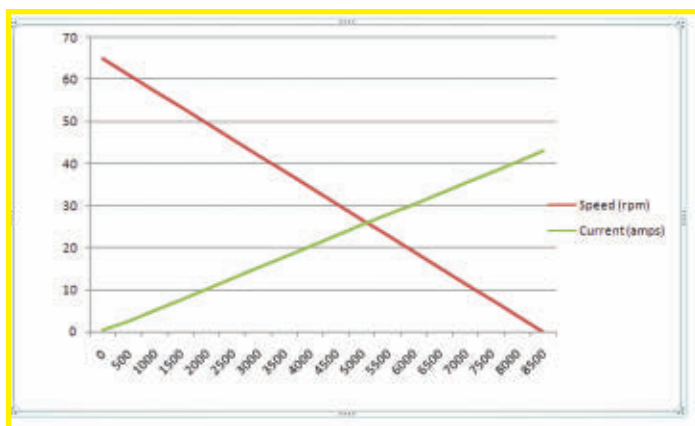
$$\frac{1.13 \text{ Nm}}{(.254 \left(\frac{m}{s^2} \right) + (9.8 \left(\frac{m}{s^2} \right) * \sin(10)) * .0762 \text{ m})} = 7.58 \text{ kg}$$

If we multiply this value by two for each drive motor and divide by two for our safety factor, then convert our kg value to pounds, we end up with a 17 pound robot. You can probably exceed this value by four or five pounds without much concern, but you know that you will not be able to build a 70 pound robot using these motors and meet your performance requirements.

A better way to look at the performance you can expect from a DC motor is the speed-torque curve. One nice thing about DC motors is that most of the relationships are linear. So, a graph that shows the relationship between speed, current, and torque is easy to develop. You only need to have a few values. These are No Load Speed, No Load Current, Stall Current, and Stall Torque. You already know the stall speed is zero.

Figure 2 is an example of the graph from a motor for a larger robot. The following values were supplied by the manufacturers specification sheet. The No Load Speed is 65 rpm, the No Load Current is .63 amps, and the Stall Torque is 8,500 oz-in. The motor has a maximum efficiency of 66%. We would like the motors to operate at about 1,000 oz-in and around 55 rpm. I was able to quickly generate a speed-torque curve in a spreadsheet. From this graph, I could see the robot will need to supply about five amps per motor. If possible, I like to stay on the lower 30% of the speed-torque curve. Stalling a DC motor can result in a much shorter operational life. If you are using a gearhead motor, it can also destroy the gearhead.

FIGURE 2. Torque oz-in, rpm graph.



Other Considerations

During the initial selection of the motor, we assumed that there was enough friction between the wheel and the surface so there is no slip. This is actually a bad assumption. Most wheels will spin when the robot is started at full speed on a slicker surface. In some cases, this can be a real problem. One solution to this situation is to select different wheels. Or, a better solution is to ramp up the motor speed instead of starting at full speed.

When selecting motors, another key consideration is the fine print on the specification sheet. A motor may be capable of supplying much more torque than the gearbox

or bearings can handle. On some gearheads, you will see a rating for overhung load. This is the maximum weight the manufacturer recommends you support from the output shaft. Another consideration is the torque limit of the gearhead. This can be less than the motor can actually supply. If you exceed this value for any length of time, you could destroy your gearhead. Better motors use better gearheads. One way to tell the quality of motor you are using is by the noise it makes when running. The louder the motors, the less efficient the gearhead motor is. High quality gearheads are very quiet, but they are also expensive.

Battery selection is another key aspect of getting the most out of your drive motors. The right battery will need to supply current for the desired robot run-time. Batteries are specified with two key values: voltage and amp-hours. An amp-hour is defined as the amount of time — in hours — the battery will supply one amp of current. To get an estimate of how long your robot will run on a given battery, just take the battery rating and divide it by the average current draw of your robot. Do not forget this should be for the entire robot, not just the drive motors. For example, my robot has a 12 amp-hr battery. My robot normally draws less than two amps, so I expect to get five to six hours of run-time with a fully charged battery. To get

a ball park of the power your batteries will need to supply to your motors, you can use the relationship between power, voltage, and current.

We've completed an exercise in basic motor sizing for a mobile robot. In most cases, you should shoot for exceeding your calculated requirements by two or three times. If you purchase undersized motors, it usually costs more than purchasing oversized motors because you will end up buying a second set of motors that are large enough to supply the power you need.

The free body diagram and basic physics is a great place to start when sizing a motor in any situation. Our example was very simple; you can always add more detail to the free body diagram. This can help you answer other questions. For example, how much of an incline can your robot handle before it tips over?

If you want more information, a great source of information is the book *Mobile Robots – Inspiration to Implementation* by Joseph L. Jones and Anita M. Flynn. The Zagros Robotics lab notes section also includes information on many topics, including DC motor selection.

(www.zagrosrobotics.com). **SV**

Tips For Selecting DC Motors

CrustCrawler

Robotics

"Build Smarter."

Smart Servos Deserve Smart Brackets!

- ✓ *Brushed, Anodized, Aluminum*
- ✓ *Integrated Pem Nuts for Easy Construction*



AX-12 Feedback

- ✓ Voltage
- ✓ Current
- ✓ Position
- ✓ Temperature



**AX Side
Bracket**



**AX Short
Bracket**



**AX 45 Degree
Bracket**



**AX Foot/
Hand Set**

See our web site for pictures, videos and more!
www.CrustCrawler.com

Or call us at:
480-577-5557

Optics

For Your Robot Using A Webcam

It is said that the eyes are the window to the soul. This leads one to wonder if we give a man-made machine our visual capability, would we then imbue our mechanical constructs with a soul? Well, I am no philosopher or spiritual statesman upon his ziggurat, but I can tell you that with a little understanding of light and optics, you can introduce a lot of capability into your robotic projects.

Machine vision is a very tricky subject with a lot of variables to understand. To the uninitiated, it is hard to know where to begin a project at all beyond buying a camera and a lens, and hoping things work out. To help demonstrate some concepts, I will detail some aspects of a project we are doing for fun here at the office. We are converting an old typewriter to "tweet" messages that are typed into it. Rather than building switch closures on every key, we thought it might be fun to use machine vision as a solution.

Before we dig too deeply into the project, there are a lot of different things to think about. I'll cover the very basics of cameras, lenses, and imaging ideas that one may encounter in most any machine vision or optical sensing project.

Light and Color

Light, as many of us know, is just another form of electromagnetic radiation. We perceive light in the wavelength range of 400 nm to 700 nm as the color range from violet to red. We do this biologically, deriving color from cones and resolution from rods in our eyes. We have color sensors that are red, green, and blue, and when we build digital devices, we generally try to make the bulk of them behave as an analogue of our own sight.

Most of today's digital cameras use red, green, and blue colored pixels (actually photosites) in a predictable arrangement called a Bayer pattern, with alternating rows being RGRGRGRG and GBGBGBGB, and so forth. Four photosites (two per row) are grouped together to form a colored pixel. Shift over one column, take another four as before, and you have your next pixel.

You will notice two things. One is that there

are two times as many green photosites as blue or red (mimicking our vision which has greater resolution in the green, and greater color sensitivity); the other is that this technique really is a misrepresentation in that the values for each derived pixel are calculated, not directly sensed. The process to make a Bayer image is called demosaicing. It is also useful to note that blue can often be the noisiest image channel.

There are alternatives to the typical camera. For instance, you can buy cameras that output raw data, or you can buy a monochrome camera. Raw data is basically the data from a color camera, but it is not convolved from photosites to pixels. Monochrome gives you the total resolution of a camera's photosites, but with no sense of color whatsoever.

So, given this information, if you need the highest resolution possible and can use a black and white camera, this has some distinct advantages, but some problems with how to understand color if you need color distinction. If you have to use a color camera and need resolution, get a raw output camera. If you just want easy to understand color images, get a color camera.

So, why use a monochromatic camera at all, if you can have a color camera. There are many reasons, actually. One is that if you can control the lighting, you can make very accurate color readings. You could use a variety of monochromatic light sources, serially exposing individual frames to different wavelengths of light and make a color camera that (while taking longer to grab an image) has a higher resolution and much higher ability to distinguish colors. This type of camera is called a hyperspectral camera.

Worth noting is that most cameras are sensitive into the 700 nm-1,000 nm range; sensitive enough, in fact, that manufacturers put little filters mounted to either the sensor or the lens. They are usually tinted blue, and if you want infrared you can carefully surgically remove them. If you have a camera that has had its infrared filter removed, you can see some very interesting effects. Plants, for instance, show bright white in infrared illumination.

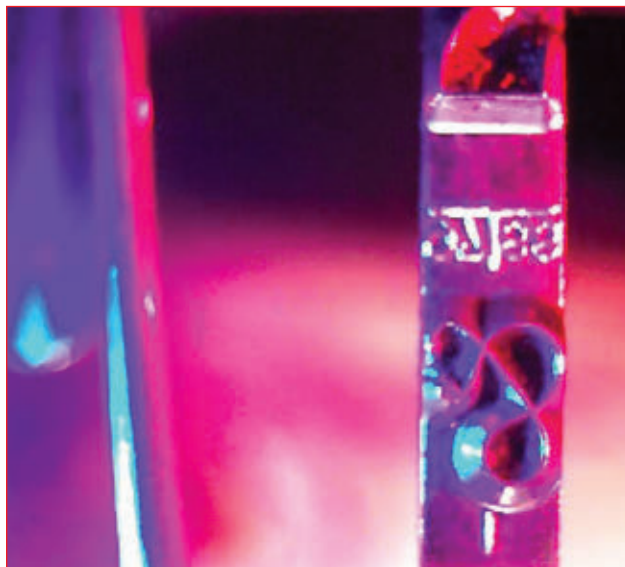
We've been thinking more about color and light, but how about the absence of light? Darkness has a lot to offer, as well. If, for instance, you wanted to know how lumpy your

automatic cookie baking machine's cookies were, we could structure the lighting to accentuate detail. A single light casting shadows across the cookie's surface would reveal its detail. In the opposite case, if you wanted to measure the homogeneity of the finished cookies, you would use a diffused light that illuminated everything very evenly. Or, how about two lights illuminating the cookie at different angles. This could be the beginnings of a gloss meter that would allow you to tell if your chocolate chips were actually melting.

Lenses

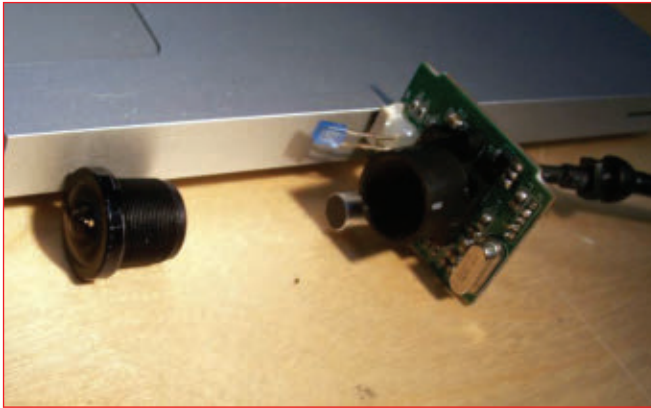
A bit of terminology is in order here. You can buy a piece of glass which has been polished on opposing sides. If both sides are flat and parallel, it is called a window. If one or both sides are curved, it is a lens element. If we group a bunch of these lens elements into a housing, we call it a lens. If you are doing image sensing, you are really better off using lenses, not lens elements. If you are simply trying to work with non-imaging light sensors, you can use lens elements. I will leave out the use of single lens elements in this discussion. It's too math-intensive and isn't any fun.

Okay, so we have light, but how do we get light onto a sensor? We use a lens! Generally, the lenses we are concerned with have some features that if understood, can allow us to have greater control of what our sensor "sees." The most dramatic aspect of visual sensor systems we can



**Typeface
illuminated
with red
and blue
LEDs.**

Board mount webcam and lens.



the area we are viewing at a given distance can be determined by:

Size= $\sin(11 \text{ degrees}) \times 3000\text{mm}$ (3m) or an area 570 mm across

Those of you that are trigonometrically inclined will note a certain symmetry here. The basics of lens design all come down to math, and a good chunk of it is geometry based.

So, back to our real world application. We are actually considering two approaches. One way would be to put retroreflective tape behind the cameras and sense when a hammer moved by sensing the reflected light behind it. This would involve a wide angle lens that would allow us to insert the camera deep into the area of the hammers, and sense them all at once.

The second approach would be to image the type-face surface as it comes in to strike. This would require a lens that was looking at a very narrow angle of view. It also requires focusing up close. There are easier ways to do this, but this gives us the opportunity to play with image processing algorithms. This is the most difficult of the two, as we have to recognize the type on the face, not where the hammer is that moved. So, we'll go with this idea.

Taking this approach, we need a camera that can image the typeface and fill the imager with as many meaningful pixels as possible (at least for now, more data may be better than less). Fortunately, I took a handful of webcams apart, and have a semi-wounded webcam with a bit of life in it. The webcam's existing lens has an IR cutoff filter installed, so if I wish to replace it, I will have to ensure we block IR.

With this camera, we are using a standard "board mount" configuration. This allows us to easily focus the lens. Focusing a lens is generally done by moving the lens further away from or closer to the imager. The closer it is to the imager, the further away it focuses, until it reaches 'infinity' focus. The opposite is also true. The further the lens is from the imager, the closer it focuses. In order to get a close-up image of the type-face and use the existing lens, I had to unscrew the lens a considerable distance.

In the world of lenses, we actually think of the lens in terms of the camera it is meant to attach to. There are different names for the different lens mounts, and they are all different styles. This scheme gives us the ability to mix/match cameras and lens manufacturers, and to swap one lens type for another.

'C' mount camera and lens.



easily affect is the angle of view of the lens. We primarily do this by selecting a lens with an appropriate focal length to the task at hand. A lens with a larger number for the focal length will have a narrower angle of view and will be called a "longer" lens. Conversely, a lens with a smaller number for focal length will have a wider angle of view and be called a "shorter" lens. The actual focal length you select will be determined by the size of the sensor you are trying to fill and the size of the object you are looking at.

Now, unfortunately it is time for some math.

Let's assume that you have a sensor with a horizontal dimension of 5 mm and a lens that has a focal length of 25 mm. You would use the following formula:

angle= $2 \times \arctan(5\text{mm}/2 \times 25\text{mm})$ or 11 degrees

Let's further assume you are using this lens and sensor to find a cola can on a surface at a robotics event where you will start three meters away from the can. We can roughly calculate that

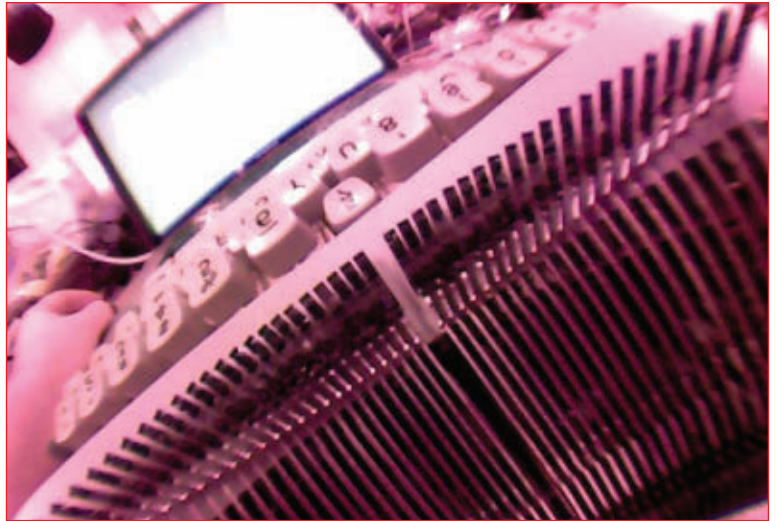
Retroreflective tape behind the typewriter's hammers clearly shows a missing one.

The easiest, inexpensive realm of cameras with the greatest flexibility is a USB or Firewire camera with what is called a "board mount" lens. UniBrain is a decent camera with a troublesome (at times) driver, but a lot of the \$10 webcams fit into the "board mount" category. These cameras generally take the same size lens (threaded with an M12x.5 thread). Since this is a common size, if you ever have to make your own mounts, you'll be able to find a tap set for this in catalogs like McMaster Carr (if your local industrial supply house doesn't carry them). Companies like Detection Dynamics carry a huge variety of lenses in different focal lengths, so you just need to keep in mind the size of the imager chip that you want to fill with your images.

The next size camera is one that has either a "C" mount or a "CS" mount. Generally, you will spend more for these lenses and the cameras they mount to. By now, you may have been looking online for different lenses and found a term called the "f#" of a lens. This is important to anyone building lenses, but until you get to this type of lens, you will not have much control over it. It is a bit complex but, f# is essentially a measure of the focal length of a lens divided by the diameter of the first element. This is a measure of the lens' ability to gather light, but it also controls how much of the area fore and aft of the object you are focusing on is in focus. The lens will most often have a ring on it which controls the iris — the thing that adjusts the amount of light entering the lens. This ring will be labeled with numbers which are the f# setting of the lens.

These C mount lenses also have a unique focusing method. The mount of the lens screws into a mount on the camera that has a flange that is a very specific distance from the sensor surface. To focus, an internal mechanism allows the focusing of the lens, and there is a ring on the outside that shows you what distance you are focused at.

Going beyond this, you get into the realm of lenses for SLR cameras, scientific imagers, and line scan imagers. These lenses are made to a higher level of quality, and can cost several orders of magnitude over board mount lenses. Line scanning is an interesting technique.



Parting Nuggets of Wisdom

When you build light sources, be mindful of using pulse width and pulse frequency to dim them. Fluorescent lights can drive webcams crazy because the flickering of the camera and the frequency of images captured interfere.

If you are imaging a fast moving subject, use a "global shutter" if you can. A "rolling" shutter captures the image row by row, which can create sheared images — a bad thing for metrology.

Shorter focal lengths create more distortion. This can be bad if you must calibrate this out before your robot's tasks can be completed.

If you are using color, make sure there is an infrared blocking filter in the system somewhere. You will saturate your sensor without it.

Get a pair of cheap plastic calipers. Use them to measure where lenses sit so you can refocus them easily.

The most important thing that I have found that makes life playing with optics easier is darkness. In a darkened space, you can add light as you need it. First, add an illuminated target — just an LED and a piece of frosted tape will work. Add some permanent ink marks and you can check focus. You can slowly introduce environmental factors like ambient illumination as your algorithms get more robust.

In closing, be methodical, take your time, and have fun. Lenses aren't voodoo; they can be easily mastered by us mere mortals. Robots are a different story. Keep them on a short leash. **SV**

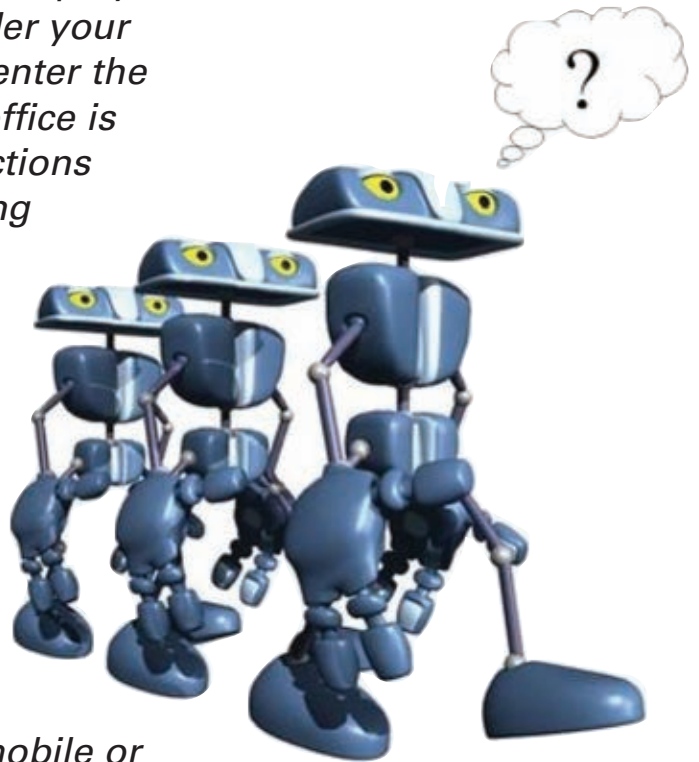
From Autonomy To Symbiosis: *Some Thoughts on the Complexities of*

NETWORKED

Robots

By Fulvio Mastrogiovanni

You have an appointment with a new employer for the long-awaited interview. Consider your actions as you try not to be late: You enter the huge building where the employer's office is located; you search for the office directions near the main hall desk, thereby finding your way towards the elevators; you wait for the elevator and take it selecting the proper floor, trying to accommodate other people as the elevator moves; you exit from the elevator, moving towards the employer's office; and so on. These are very simple actions for humans, to such an extent that you are not thinking about them as you perform them. Strictly speaking, they are so "simple" that no multi-billion-dollar mobile or humanoid robot would be able to manage the whole process.





Consider the simple action of selecting the floor once you are in the elevator. For robots (and for humans as well, but we are not aware of this), this would require the ability to recognize the button to press for the proper floor using some sort of sensing device (with high probability, a camera). There could be problems related to occlusions and changing lighting conditions to deal with, plus, there needs to be some sort of dexterous manipulation device (arms for humans and similar mechatronic structures for robots) to press the button with, and so on. These are very hard problems in robotics. Although examples of mobile robots which are — in principle — able to achieve these results have been presented in the market (see **Figure 1**), vision is still unsuitable to allow robots to robustly understand the surrounding environment under a cognitive perspective. Furthermore, given a robot (i.e., an algorithm operating on noisy sensory data) able to purposively scan for or to correctly press buttons within elevators, we are not guaranteed that these operations are successful in each possible scenario. Shape, color, and dimension of buttons change from elevator to elevator, and there could be a lot of people around. These issues are just two examples of the whole process; navigation, searching for the right floor without a map, and interacting with people are subject to ongoing and active research. Each one of these is a sub-field of robotics in its own right.

In spite of their limitations, mobile or humanoid robots are going to be commercialized very soon. During the past few years, robots have successfully operated in structured environments (like the robot in **Figure 2**), on Mars, and in abandoned mines and undersea, searching for lost treasures. Therefore, there is a lot of work for robots. They just can't have interviews or sign the contracts yet.

Design Methodologies in Mobile Robotics

Currently, there are mainstream trends adopted to deal with these fundamental problems in robotics, which are metaphorically based on strong and harsh philosophical positions:

Conservatism. Robots are exploited in very "simple" scenarios where environments are almost static and can be easily understood, thus requiring neither complex cognitive capabilities nor sophisticated purposive interaction with the world.

Although a lot of effort is currently being dedicated for improving their autonomy, Mars rovers are almost tele-operated. Commands are issued from Earth on

Figure 1.



the basis of images and other data provided by the rover on Mars (see article in the January '09 issue of *SERVO Magazine*). The same is true for robots exploring undersea or abandoned mines. Industrial robots and UGVs (Unmanned Guided Vehicles) work pretty well in industrial and controlled scenarios. No people around, no changes in the environment, and so on.

Idealism. The ultimate goal of robotics is to introduce autonomous robots into the real world, to push research towards robots which are more and more similar to humans in their capabilities, thus investing efforts and dollars in actual research. This goal is still far from being achieved. Although robots exist that are able to map and navigate in very large environments, to avoid obstacles, and to travel very long distances (like the 60+ miles travelled in the DARPA Urban Challenge in 2007; see [1]), if you think about it, these are impressive advances pushing the limits of the automotive industry, rather than robotics. No robot exists that is able to leave a room in a building, to change floors using either the stairs (in the case of humanoids) or an elevator; and then reach another room autonomously or without human intervention.

Pragmatism. To accept our own limits is the first step towards self-improvement. This is true for robots, as well. Several research groups are accepting robot's intrinsic limitations (at least at the current status of research) and are working towards providing



Figure 2.



robots with alternative solutions; a sort of intermediate position between conservatism and idealism. The IEEE gave rise to a Technical Committee devoted to discuss these issues [2]. These lines of research are mainly governed by industrial needs which require the automation of large parts of industrial workflow outside the normal scope of controlled scenarios: surveillance in indoor and outdoor areas; objects transportation and goods delivery in (semi)structured, human populated scenarios; and so on.

Robots and Intelligent Environments

Networked robots represent the most promising alternative solution to many failures exhibited by completely autonomous robots during the past few years. Roughly speaking, networked robots are robots that are connected to parts of their environment (namely intelligent devices) or robots that are able to cooperate, sense, and act with the help of their own environment. This is true to such an extent that a real symbiosis is foreseen between the mobile or humanoid robot and its surrounding environment. The individuality of a single device is somewhat lost to gain a more powerful network of individuals.

A number of explanations are mandatory here. First, we consider indifferently both humanoids and wheeled mobile robots (i.e., we abstract from specific locomotion, sensing, and navigation capabilities, which are, of course, important but not relevant here), assuming that robots are provided with some computational capabilities on-board, or at least have a Wi-Fi Ethernet network connection (the common 802.11g or similar). This is reasonable, since nowadays robots mainly exploit PC104-like boards or similar computing machinery. Second, intelligent devices are sensors and actuators provided with controllers able to acquire data or to issue commands. These can be cameras, infrared, temperature or gas sensors, or controllers able to operate automated windows, doors, or elevators, to name a few. These devices are logically and architecturally part of a network (which can be a fieldbus network, commonly used in industrial automation, such as Echelon LonWorks or CanBus; or wireless, such as Bluetooth, ZigBee, or the Wi-Fi Ethernet). Third, the robot and the intelligent devices are all part of the same network, therefore able to communicate with each other and with other workstations within the network itself. Either centralized or decentralized algorithms exist to coordinate the activities of all the entities (robots and intelligent devices) which are part of the network.

Networked robots are possible due to the recent advances in many related technological fields: robotics itself; ambient intelligence (see article in the December '08 issue of *SERVO Magazine*); and distributed sensor networks. However, it is very difficult to have the expertise on both mobile robotics and ambient intelligence. Therefore, a lot of early work in this field has been done in simulation [3]. We can expect that as soon as these fields improve and

produce novel and astonishing technology, networked robots will expand their scope in an impressive manner.

Case Study: Storage and Warehouses

A networked mobile robot is requested to manage retrieval and delivery of objects on demand within a huge storage facility. In this case, objects are essentially wood boxes which need to be delivered upon user requests at known locations. Boxes are located on multiple floors of the facility, thereby requiring the mobile robot to use lifts to reach the correct area where the box is located. Each box is provided with one or more RFID (Radio Frequency Identification) tags; this allows a quick retrieval of information from the box itself (e.g., its content type). A mechanical automated device must be realized to bring boxes from their location in order to load them on the robot. This is already available in modern industrial automation. The location of each box (for example, composed by floor, area, and position within the storage block) is maintained within a central database that is continuously updated according to the on-going storage activity.

As soon as a new box request is issued, several things happen. It is worth noting that box retrieval and delivery is a complex issue that requires several actions to be performed in a given sequence. These actions are planned in a central workstation which is part of the whole network of devices. The plan is then uploaded onto the networked robot, which starts its execution. During this execution, the robot's and devices' actions are continuously monitored in order to infer faults in the plan and to react accordingly.

Assuming everything goes smoothly, the robot starts navigating towards the area where the box is located. Each automated door it encounters during navigation is managed by an intelligent device, which is responsible to open/close the door upon receiving the right "credentials." As soon as the robot approaches a door, it requests the door to open by providing its own credential (its own ID). If it is allowed

Resources

[1] Tartan Racing

www.tartanracing.org

This is the website of the Carnegie-Mellon based team which won the DARPA Urban Challenge in 2007.

[2] IEEE Technical Committee on Networked Robots

<http://faculty.cs.tamu.edu/dzsong/tc/index.html>

This is the website where all the researchers around the world interested in networked robotics share their opinions.

[3] Player/Stage

<http://playerstage.sourceforge.net>

This is a mobile robotics simulator which allows experimenting with networked applications.



to enter the area, the door will open. Otherwise, a fault will be reported in the overall plan. If the robot switches floors, a similar mechanism has to be in place for lifts. Each lift that must be used by the robot is an intelligent lift — one that is able to exchange information with the robot itself about issues related to “lift present at the robot floor,” “doors opened,” “at the requested floor,” and so on. Wi-Fi connections are fundamental here. The robot is able to request the lift, to select the floor, and to enter and exit from the lift without being able to recognize and press buttons.

Once the robot reaches the correct area for box loading, a complex interaction starts with the proper networked mechatronic device responsible for picking up the boxes from the shelves and loading them on the robot. After the docking phase, the robot requests the proper box by exchanging information (something akin to the box ID) with the relevant intelligent device. Once the box is loaded, the robot navigates towards the final destination to accomplish the delivery. In order to immediately infer faults in the overall plan, the intelligent devices continuously track the mobile robot and report their actions to the centralized supervisor. This can be a simple mechanism tracking RFID readings along the robot path and checking if these match the information provided by the robot itself.

Conclusion

Let's come back to robots searching for work. The humanoid D2Q9 has an important appointment with the new employer for the long-awaited interview at the NASA offices. D2Q9 tries not to be late. It enters the huge building where the employer's office is located through the main automated door which is queried through Wi-Fi connection. The door searches for the appointment in a centralized database, thus allowing D2Q9 to enter. D2Q9 searches for the right directions near the main hall desk, again querying a database which maintains knowledge about the office locations.

Once provided with the correct floor and directions, D2Q9 is able to find its way towards the elevators. D2Q9 calls the elevator at the current floor through a silent, invisible Wi-Fi request. The elevator acknowledges D2Q9 about the request and, after some time, the elevator's door opens. D2Q9 is notified by the elevator itself that it's open. D2Q9 takes the elevator with other people. It selects the floor without pressing any button, but by making a request to the elevator. Once at the correct floor, D2Q9 exits from the elevator, moving towards the employer's office.

Now it's up to D2Q9 to pass the interview ... **SV**



AP CIRCUITS
PCB Fabrication Since 1984

As low as...

\$9.95
each!

Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

www.apcircuits.com



Extreme Robot Speed Control!

Sidewinder

\$399

- ◆ 14V - 50V - Dual **80A** H-bridges - 150A+ Peak!
- ◆ Adjustable current limiting
- ◆ Temperature limiting
- ◆ Three R/C inputs - serial option
- ◆ Many mixing options - Flipped Bot Input
- ◆ Rugged extruded Aluminum case
- ◆ 4.25" x 3.23" x 1.1"

RC Control

\$39.99

Scorpion Mini
◆ 2.5A (6A pk) H-bridge
◆ 5V - 20V
◆ 1.6" x .625" x 0.25"

\$159.99

Scorpion XXL
◆ Dual **20A** H-bridge 45A Peak!
◆ 5V - 28V
◆ 2.7" x 1.6" x 0.75"

BotsIQ Favorite!

\$104.99
2+ price

Scorpion XL
◆ Dual **13A** H-bridge
◆ 5V - 28V
◆ 2.7" x 1.6" x 0.5"

Dalf Motion Control System

- ◆ Closed-loop control of two motors
- ◆ Full PID position/velocity loop
- ◆ Trapezoidal path generator
- ◆ Windows GUI for all features
- ◆ Giant Servo Model!
- ◆ C source for routines provided
- ◆ PIC18F6722 CPU

\$250



See www.embeddedelectronics.net

NEW!

Magnum775

- ◆ planetary gearbox
- ◆ 20:1 ratio - 700 rpm
- ◆ RS-775 motor
- ◆ Nearly 700W!
- ◆ Build something - rule BotsIQ!

\$89

H-bridges: Use with Dalf or Stamp



\$79

Simple-H

- ◆ 6-28V **25A!**
- ◆ 2.25"x2.5"x0.5"
- ◆ 3 wire interface
- ◆ current & temp protection



We also do consulting!
Give us a call for a custom motor control to meet your exact needs

www.robotpower.com

Phone: 253-843-2504 • sales@robotpower.com

DIGILENT'S

Robotic Starter Kit

There are times within a project that I spend as much time using a screwdriver and wrench as I do soldering and coding with a C compiler. I'm a coder by nature, so I'd much rather do my robotic work with a keyboard and microcontroller. Occasionally, I do break that orbit to descend upon and service the necessary mechanical aspects of a robotic project. After all, I'm no stranger to hand and power tools.

If you've read this far with me, we're most likely in a synchronous orbit. So, let's fire our descent thrusters and break out the screwdrivers, wrenches, pliers, and wire cutters. Our landing party has a robotic device to code *and* assemble.

We won't be pulling a Mars rover out of the box you see in **Photo 1**. However, there is enough mechanical and electrical hardware within the box to build up a pretty smart little mechanical animal.

Most of my robotic projects don't use wheels and don't

go scooting about. The Digilent Robotic Starter Kit includes a set of wheels but that doesn't mean we have to use them. There are no native gear trains in the Digilent kit box but there are a couple of 1:19 ratio motor/gearbox drives in there. If your idea of applying the Starter Kit doesn't involve rolling around on the floor, simply replace the wheels that you would normally drive with the included motor/gearboxes with gears in your own mechatronic design. After all, your build does not have to be a dedicated 1950's B movie style robot; it can be any "robotic" device. The wheels have other possibilities, as well. For instance, your robotic device may use the rubber treaded wheels as friction drives for a belt or drum assembly.

My Starter Kit box contained the mechanical parts you see in **Photo 2**. It's rather obvious that this collection of mechanical parts is aimed at a rolling robotic device. I put a screwdriver to the metal, bolts, and screws to fashion the default peg-leg floor scraper you see in **Photo 3**. Remember this is a "STARTER" kit so the idea is to get you up and



PHOTO 1. Once we sort out the robotic support hardware, there are still plenty of electronic goodies in this box to play with.

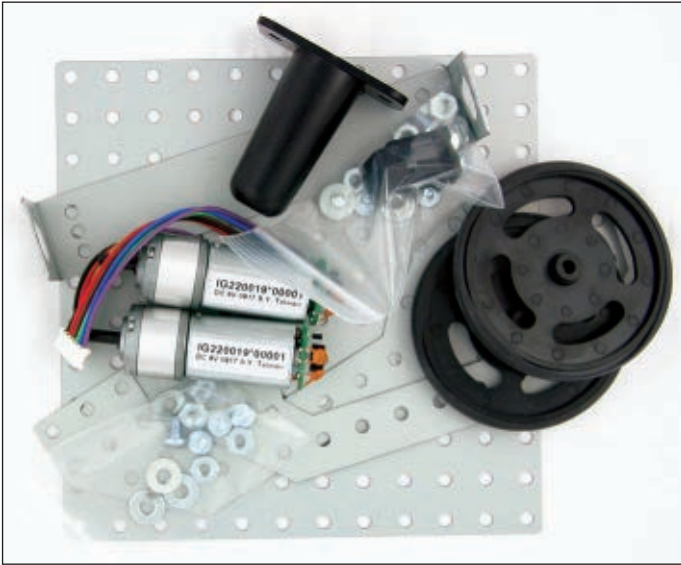


PHOTO 2. A rolling Digilent Robotic Starter Kit variant can be assembled using the base plate, support post, wheels, dual motor mount bracket, and the pair of motor/gearboxes shown here. There are also a couple of expansion plates in this shot that you can bolt on.

robotic as quickly as possible with the least amount of effort. I decided to keep things visually simple by not installing the bulk of the wires and cables.

In last month's Digilent Cerebot discussion, I demonstrated via C source code how to turn the motor/gearbox shafts using the Cerebot 32MX4 electronics. I also have an updated version of the factory firmware that I will post on the *SERVO* ftp site for you. The Starter Kit factory firmware provides various 32-bit PIC32MX motor/gearbox routines that you can use to model your own firmware motor/gearbox drive train. In addition to core motor/gearbox code, the Digilent motor driver firmware also contains routines that operate against the motor/gearbox rotational sensors. Now that we've accomplished the hardware assembly task, let's pull some more stuff out of the box. I've got my eye on that joystick Pmod.

Writing A PmodJSTK Driver

I had so many Digilent goodies to play with that I failed to take the time to completely read through the PmodJSTK datasheet. Take a look at **Photo 4**. What you don't see in the shot is an AVR ATtiny24 microcontroller that is monitoring the position of the X and Y axis potentiometers and sensing the states of a trio of pushbutton switches. The PmodJSTK's little AVR microcontroller also supports a pair of user-accessible LEDs.

The ATtiny24 is just that — tiny. So, my first thoughts were that its SPI Slave node implementation was done using a software bit bang algorithm. To my surprise, the ATtiny24 contains what Atmel calls a Universal Serial

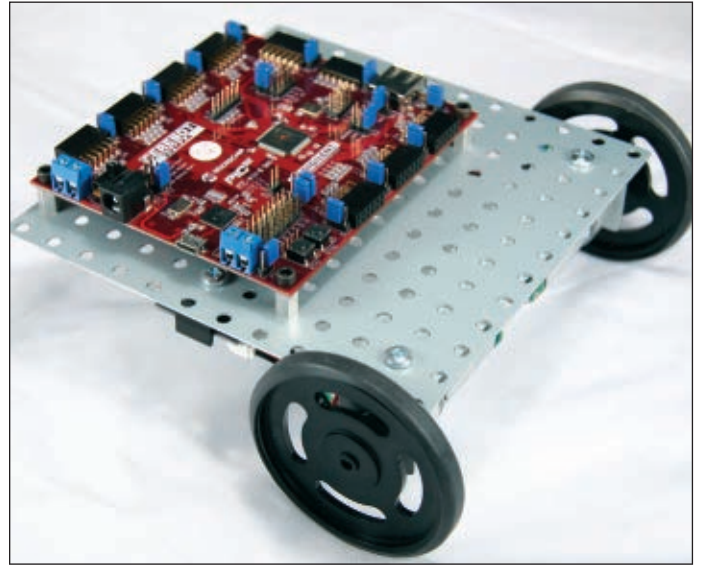
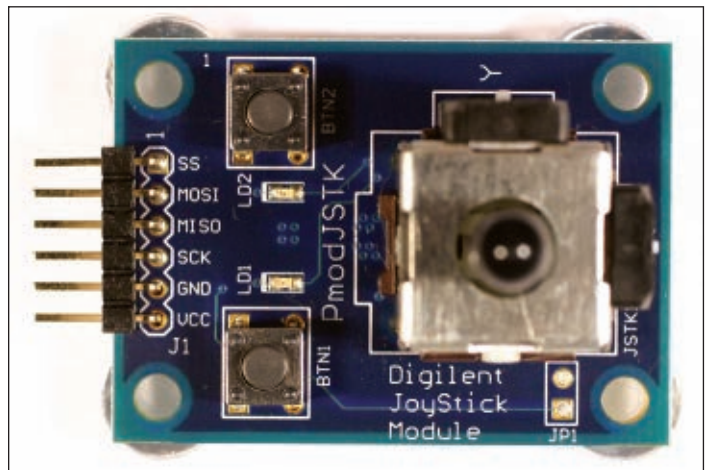


PHOTO 3. As a *SERVO* reader and robotic enthusiast, this is very simple for you. However, keep in mind that this is a "STARTER" kit. With that thought, the engineers at Digilent provide a step-by-step assembly description in the Starter Kit Reference Manual, which is available free from the Digilent website.

Interface which is capable of what Atmel describes as three-wire mode. As it turns out, Atmel's three-wire mode is actually SPI to everyone else.

The SPI (Serial Peripheral Interface) protocol is a Motorola invention. A Master SPI portal consists of an input, an output, a clock output, and an optional slave select output. Slave SPI portals are also fitted with an input and an output. However, the Slave's clock signal originates at the Master node and the Slave Select is an input used by

PHOTO 4. I didn't have to do any fancy coding to get this PmodJSTK to talk to me. The Microchip PIC32MX C compiler's SPI API library routines took all of the complexity out of SPI driver firmware. I also wrote a PmodJSTK driver that does not use the API which is embedded in this month's text.



the Master node to enable the Slave node for a communications session. Slave Select (SS) is normally an active low signal.

Determining how to interconnect SPI Master nodes to SPI Slave nodes can be confusing as Atmel and Microchip don't use the same SPI nomenclature. For instance, Atmel uses *DI* and *DO* to respectively describe the SPI portal's input and output functions. Microchip changes Atmel's *DI* naming convention to *SDI* and Atmel's *DO* is morphed to *SDO* in Microchip land.

A read of the ATtiny24's datasheet reveals that the Microchip clock signal *SCK* is translated to *USCK* on the Atmel ATtiny24 side. All of this SPI I/O confusion is eliminated by associating the Atmel and Microchip SPI nomenclature to the silkscreened SPI identifiers you see on the PmodJSTK in **Photo 4**.

The PmodJSTK's SPI portal descriptors are of Motorola origin. MOSI is short for Master Out Slave In. Thus, from the Master node's point of view, the *MOSI* pin is the *DO* or *SDO* output pin that connects to the Slave node's SPI input pin. Conversely, MISO identifies the Master In Slave Out pin, which directly relates to the Master node's *DI* or *SDI* pins (which will connect to the Slave node's output pin). These verbose SPI portal pin names also work perfectly to accurately describe the Slave node's SPI portal. *MOSI* from the Slave node's perspective is a *DI* or *SDI* input pin connected to a Master node output, while the Slave node's *MISO* pin will perform data output duty for a Master node input.

The PmodJSTK's ATtiny24 microcontroller is programmed to operate in SPI Mode 0. SPI Mode 0 is defined as the SPI *SCK* signal idling at the logical low level. An *SCK* transition from logically low to logically high forces the input data to be sampled. In Mode 0, output data on the node's output pin is changed on the falling edge of the *SCK* cycle. Thus, the SPI protocol allows the transfer of a bit in and a bit out at each end of a single *SCK* cycle.

The SPI data stream used by the PmodJSTK consists of five bytes. The first byte of the five-byte data package shifted into the PmodJSTK by the Master node controls the PmodJSTK's LED pair. To control the PmodJSTK LEDs, the most significant bit of the first byte shifted into the PmodJSTK must be set. The illumination state of the PmodJSTK LED pair is determined by the least significant bits of the first byte shifted into the PmodJSTK. If the least significant bits of the first shifted in byte are zero, both LEDs are extinguished. A set bit in either of the least significant bits of the first shifted in byte will illuminate the respective LED. For instance, to illuminate both LEDs, we set both of the least significant bits (0b10000011) in the first shifted in byte. Setting the least significant bit of the first shifted in byte will illuminate LED1 (0b10000001), while setting bit 1 only will force visible light from LED2 (0b10000010).

The PmodJSTK ignores the next four bytes that are transmitted by the Master node. The PmodJSTK uses the remaining four bytes to clock out the joystick X and Y axis position numbers. In addition to that tricky Universal Serial Interface, the ATtiny24 microcontroller manages to squeeze in an on-chip analog-to-digital (A-to-D) converter, which is used to measure the voltage at the joystick's pair of potentiometer wiper pins. The joystick potentiometers are tied between the voltage rail of the PmodJSTK. The ATtiny24's A-to-D converter engine resolves the wiper voltages to 10 bits (0bxx11111111 or 0x03FF). The 10 bits are assembled into a 16-bit package and transmitted via the PmodJSTK's SPI portal as a low and high byte. The two most significant bits of the upper A-to-D converter byte are always cleared. The first byte clocked out of the PmodJSTK is the least significant byte of the X A-to-D measurement value. The least significant byte of the X axis value is clocked out as the Master node clocks in the LED control byte. The next three bytes clocked out of the PmodJSTK are the most significant byte of the X axis value, the least significant byte of the Y axis value, and the most significant byte of the Y axis value.

The fifth and final byte of the clocked out data package contains the status of the PmodJSTK's pushbuttons. The joystick is also equipped with a switch which is activated by depressing the joystick handle. The least significant bit of the clocked out fifth byte is set when the joystick switch is engaged. Depressing BTN1 returns a binary bit pattern of 0b00000010 and BTN2 announces a depressed state by setting bit 2 of the clocked out fifth byte (0b00000100).

Now that you have an understanding of what the PmodJSTK is capable of, let's assemble some code according to what we know. There are two sure-fire ways to code the PmodJSTK driver. The old-fashioned way is to write the code using direct register manipulation. There is also an API available for the PIC32MX460F512L SPI engine. We'll go with the old school method here and I'll post the API code for you on the *SERVO* website.

A Basic C PmodJSTK Driver

If you happened to miss our previous Cerebot 32MX4 discussion, we traversed these PIC32MX460F512L fuse settings to obtain some important clock and timing information:

```
/* *****  
/* DIGILENT JOYSTICK DRIVER MANUAL REGISTER  
/* VERSION  
/* LAST UPDATED 11/16/2009  
/* FIRMWARE BY FRED EADY FOR SERVO MAGAZINE  
/* CHANGES/ADDITIONS  
/* *****  
/* THIS CODE WRITTEN AND COMPILED USING  
/* MICROCHIP C32 V1.10
```

```
#include <p32xxxx.h>
#include "stdtypes.h"

#pragma config ICESEL    = ICS_PGx2
    // ICE/ICD Comm Channel Select
#pragma config BWP       = OFF
    // Boot Flash Write Protect
#pragma config CP        = OFF
    // Code Protect
#pragma config FNOSC     = PRIPLL
    // Oscillator Selection
#pragma config FSOSCEN   = OFF
    // Secondary Oscillator Enable
#pragma config IESO      = OFF
    // Internal/External Switch-over
#pragma config POSCMOD   = HS
    // Primary Oscillator
#pragma config OSCIOFNC  = OFF
    // CLKO Enable
#pragma config FPBDIV    = DIV_8
    // Peripheral Clock divisor
#pragma config FCKSM     = CSDCMD
    // Clock Switching & Fail
    // Safe Clock Monitor
#pragma config WDTPS     = PS1
    // Watchdog Timer Postscale
#pragma config FWDTEN    = OFF
    // Watchdog Timer
#pragma config FPLLDIV   = DIV_2
    // PLL Input Divider
#pragma config FPLLMUL   = MUL_16
    // PLL Multiplier
#pragma config UPLLDIV   = DIV_2
    // USB PLL Input Divider
#pragma config UPLLEN    = OFF
    // USB PLL Enabled
#pragma config FPLLODIV  = DIV_1
    // PLL Output Divider
#pragma config PWP       = OFF
    // Program Flash Write Protect
#pragma config DEBUG     = OFF
    // Debugger Enable/Disable
```

What you need to come away with from these fuse settings is that the PIC32MX460F512L's CPU is being clocked at 64 MHz and the peripherals are using an 8 MHz clock, which is derived from the CPU clock. The 32MX4's PIC32MX460F512L is being clocked externally by an 8 MHz crystal. So, the input to the PIC's PLL is 4 MHz, which is determined by the configuration fuse entry FPLLDIV = DIV_2. The PLL Multiplier is 16 and the PLL Output Divider is 1. Doing the simple 16x multiplication puts the output of the PLL at 64 MHz which is the CPU clock. The PLL output also feeds the Peripheral Clock which is programmed to provide an output of 64 MHz divided by 8, which gives us a Peripheral Clock of 8 MHz.

Recall that we will need to store five bytes of clocked in data from the PmodJSTK. We'll create a simple five-byte array for this purpose:

```
BYTE spDataIN[5];
    //5-byte data buffer
```

```
BYTE spIndex;
    //data buffer index
```

The *spIndex* byte is used as a pointer to the bytes of the five-byte *spDataIN* array. If you're wondering what the *sp* prefix represents, it's just a way that we can immediately identify what the variable is used for. I've put together a table to explain the domains of the various prefixes we'll be employing:

```
/** *****
/* sp -> SPI reference
/* p  -> I/O pin reference
/* m  -> macro reference
/* b  -> bit reference
/** *****
```

From the table you can see that a variable beginning with a *b* is referencing a bit while a variable prefixed by a *p* is representing a reference to an I/O pin. In the code that follows, *bspON* (read as bit SPI ON) is a human-readable name for the 15th bit in the SPI1CON register. Interpret the *pjoystick* definition name as the I/O *pin* (*p*) that enables the PmodJSTK:

```
#define pjoystick      9
    //slave select at RD9 of MASTER
#define mjoystickOFF   PORTDSET =
    (1 << pjoystick);
#define mjoystickON    PORTDCLR =
    (1 << pjoystick);

#define bspON          15    //SPI ON bit
#define mspENBL        SPI1CONSET = (1 << bspON);
#define mspDIS         SPI1CONCLR = (1 << bspON);
```

Note that we're using the PIC32MX460F512L's atomic bit operators to set and clear an I/O pin (RD9) and manipulate a bit within a PIC Special Function Register (SPI1CON). Now that we have our SFR and I/O macros in place, we can write our SPI initialization code.

The first order of SPI business we must attend to is to clear the status registers associated with sending and receiving SPI data and preload the SPI configuration registers to reflect our desired mode of operation. We'll do this by creating a function called *init*:

```
void init(void)
{
    TRISDCLR = (1 << pjoystick);
    //RD9 is output pin
    SPI1CON = 0x0000;
    //stop and reset SPI engine
    spDataIN[0] = SPI1BUF
    //clear SPI receive buffer
    SPI1BRG = 7;
    //500KHz SPI clock
    SPI1STATCLR = 0x40;
    //clear SPIROV
    SPI1CON = 0x00020;
    //8 bit-Mode 0-Master Node
    mspENBL;
    //enable the SPI engine
```

```

mjoystickON;
    //select slave
    for(y=0;y<0x0300;++y)
        ++z;
    //delay for slave select
    x = 0x83;
    //illuminate both LEDs
}

```

The atomic clear operation we let loose on the TRISD register set the direction of I/O pin RD9 to input. Writing a zero (0x0000) to the SPI1CON register guarantees that the ON bit is cleared which stops the SPI engine in its tracks. The zeroing of the SPI1CON register doesn't clear the data buffer SPI1BUF. So, we must perform a read against SPI1BUF to clear it. With the SPI engine stopped, we can configure it for our purposes.

The PmodJSTK datasheet sets a 1 MHz SPI SCK limit for proper operation. To be safe, we'll run our SPI SCK signal at 500 kHz. Here's the math behind determining what to load into the SPI1BRG register:

$$F_{SCK} = F_{PB} / 2 * (SPI1BRG+1)$$

Where:

$$F_{SCK} = 500 \text{ kHz}$$

$$F_{PB} = 8 \text{ MHz}$$

Doing some quick head math tells us that we need to divide the 8 MHz Peripheral Clock (F_{PB}) by 16 to obtain the 500 kHz F_{SCK} value. So, we load SPI1BRG with a value of 7. If you decide to run at the max SPI SCK rate, you'll want to divide F_{PB} by 8, which means you'll have to load 3 into SPI1BRG.

Clearing the SPIROV overflow bit infers that we will be checking for overflow errors in our code. Let's keep it simple and not complicate our driver with error checking until we're sure we have some solid code under our belts. However, it's a good idea to go ahead and clear the overflow bit as the SPIROV bit is not automatically serviced by the PIC hardware and can only be cleared manually in this manner. Note we used the SPI1STAT register's atomic bit clearing apparatus to clear the SPIROV bit.

Virtual	Symbol Name	Value
A000_0100	= spDataIN	
A000_0100	[0]	0xBB
A000_0101	[1]	0x02
A000_0102	[2]	0x19
A000_0103	[3]	0x02
A000_0104	[4]	0x01
A000_0105	spIndex	0x02
A000_00FC	x	0x81
BF80_5810	SPI1STAT	0x00000008
BF80_5830	SPI1BRG	0x00000007
BF80_5800	SPI1CON	0x00008020
BF80_5820	SPI1BUF	0x00000002

All we have to do is set the Master Mode Enable bit in SPI1CON for eight-bit Mode 0 operation. At this point, the SPI1BUF is clear, the SCK clock rate is set, and all of the SPI status bits are initialized. We're ready to start the SPI engine using the *mSPENBL* macro as the key. Once the SPI motor is running, we can release the brakes. The PmodJSTK requires that its Slave Select line be driven low before any communications sessions can be established. The brake release in this case is the macro that drives the PmodJSTK's Slave Select line logically low, *mjoystickON*.

The PmodJSTK datasheet recommends that a delay of at least 15 μ s be observed following the activation of the PmodJSTK's SPI portal before beginning data transmission. The little *for* loop I threw in as a 15 μ s delay isn't very scientific but it works. We're using the PmodJSTK's LEDs as activity indicators. So, I initially illuminate both LEDs by clocking in the first byte with a value of 0x83 to give us an immediate visual indicator of the PmodJSTK's operation. Now that our SPI machine is under way, let's examine the code that moves the data:

```

do{
    for(spIndex = 0; spIndex < 5; ++spIndex)
    {
        for(y=0;y<0x0300;++y)
            //red neck delay for slave
            //select
        {
            ++z;
        }
        if(SPI1STATbits.SPITBE)
            //wait until Transmit
            //Buffer Empty
        {
            SPI1BUF = x;
            //send variable x
        }
        while(!SPI1STATbits.SPIRBF);
        //wait for Receive Buffer
        //Full
        spDataIN[spIndex] = SPI1BUF;
        //read the clocked in data
    }
    mjoystickOFF;

    //deselect the slave
    if(++x > 0x83)
        //count 0x80 - 0x83 then
        //reload
    {
        x = 0x80;
        //extinguish the LEDs
    }
    mjoystickON;
    //enable SS line
}while(1);

```

The PmodJSTK datasheet states that we should

SCREENSHOT 1. The bytes contained within the *spDataIN* array are a result of my holding the joystick in an off-center position while depressing the joystick handle to activate the joystick switch.

regulate the message activity with a minimum of 10 μ S between transmissions. So, I threw in my red neck delay routine which runs just before each SPI communications cycle. When the PIC32MX460F512L's SPI transmit buffer is empty, we can load it with data to be clocked out. The SPI engine automatically transfers the data out while clocking in data from the PmodJSTK. We wait for the incoming data to fill the SPI receive buffer before performing a receive buffer read which clears the receive buffer. At the end of five SPI cycles, we have five bytes in our *spDataIN* array that we can work with. **Screenshot 1** is the result of my holding the joystick off-center and depressing the joystick switch. Let's decipher the contents of the *spDataIN* array values in **Screenshot 1**. The X axis value is 0x02BB, the Y axis value is 0x0219, and a 0x01 in the fifth byte signifies that the joystick switch is depressed. At the time I stopped to capture the aforementioned values, LED1 was illuminated (x = 0x81).

Roll or Control

The PmodJSTK and the firmware driver we just

Fred Eady may be reached via email fred@edtp.com and the EDTP Electronics website: www.edtp.com

completed puts a joystick, three momentary pushbuttons, and a pair of LEDs at your disposal. If you take in consideration what we accomplished last month combined with the work we performed this month, you can roll and/or control all things robotic using the Diligent Cerebot 32MX4/Diligent Robotic Starter Kit combination. **SV**

Sources

Diligent Cerebot 32MX4
Diligent Robotic Starter Kit
PmodJSTK
Diligent
www.diligentinc.com

MPLAB IDE
MPLAB C32 Compiler for PIC32 Microcontrollers
PIC32MX460F512L
Microchip
www.microchip.com

Robotics Showcase



CATCAN
www.catcan.com.tw

CATCAN SMART SENSOR LITE

PCB dimension : 56mm x 36mm (38.9 mm x 16 mm)
Working voltage : 4.8V-10V
Interface : I²C
I²C Bandwidth : 100KHz - 400KHz
Signal filter : integrated CATCAN motion filter
Data reported :
3D vector (X/Y/Z) : $\pm 180^\circ$
3D G sensor raw data (X/Y/Z) :
 $\pm 2G/\pm 4G/\pm 8G/\pm 16G$ (10bit/11bit/12bit/13bit)
3D magnetic sensor raw data (X/Y/Z) : ± 2 Gauss (12bit)
Z axis gyro sensor raw : $-300 - 300^\circ/\text{sec}$ (12bit)

These products are not connected to or endorsed by The LEGO Group. LEGO, Mindstorms, NXT, RCX and RIS are trademarks of The LEGO Group.

GFEC 茂綸股份有限公司
GALAXY FAR EAST CORP.
www.gfec.com.tw
TEL: 886-2-8913-2200 FAX: 886-2-8913-2277
E-mail: service@catcan.com.tw



ALL ELECTRONICS CORPORATION

THOUSANDS OF ELECTRONIC PARTS AND SUPPLIES

VISIT OUR ONLINE STORE AT
www.allelectronics.com

WALL TRANSFORMERS, ALARMS, FUSES, CABLE TIES, RELAYS, OPTO ELECTRONICS, KNOBS, VIDEO ACCESSORIES, SIRENS, SOLDER ACCESSORIES, MOTORS, DIODES, HEAT SINKS, CAPACITORS, CHOKES, TOOLS, FASTENERS, TERMINAL STRIPS, CRIMP CONNECTORS, L.E.D.S., DISPLAYS, FANS, BREAD-BOARDS, RESISTORS, SOLAR CELLS, BUZZERS, BATTERIES, MAGNETS, CAMERAS, DC-DC CONVERTERS, HEADPHONES, LAMPS, PANEL METERS, SWITCHES, SPEAKERS, PELTIER DEVICES, and much more....

ORDER TOLL FREE
1-800-826-5432
Ask for our **FREE 96 page catalog**

Recycling & Remarketing High Technology

WEIRDSTUFF® WAREHOUSE

Software, Computers, Electronics, Equipment, Doo-hickies

384 W. Caribbean Dr. Sunnyvale, CA 94089
Mon-Sat: 9:30-6:00 Sun: 11:00-5:00
(408)743-5650 Store x324

 **WE BUY AND SELL EXCESS & OBSOLETE INVENTORIES!**

FREE COMPUTER RECYCLING
We recycle computers, monitors, and electronic equipment. M-Sat 9:30-4:00 

 **WEEKLY SEALED BID SALE AUCTION**
Hi-tech items, electronics test equipment, and more!

GIANT AS-IS SECTION
10,000 sq. ft. of computers, electronics, software, doo-hickies, cables, and more! 

also check out our...

eBay Store
stores.ebay.com/WeirdStuff-Inc

WWW.WEIRDSTUFF.COM

Self Similar Optimization

This new coding idea will change how you design and optimize robots.

"Genetic Algorithms" have become commonplace in industrial, commercial, and robotic design processes. They have even been used to program robot brains. Now there is an alternative — a new kid on the block called Self Similar Optimization. With just a few lines of computer code it allows you to sometimes equal or surpass what has been done with genetic algorithms.

A Self Similar Curve

The self similar curve used in the calculations is one of the simplest available. It is just the exponential function found in most computer languages. The input to the exponential function is going to be a number ranging from zero to some negative value known as the precision; for example, from 0 to -4. This curve is shown in **Figure 1**.

It is not altogether clear that the curve is self similar, however, looking at **Figure 2** you can see that the curve is made up of smaller and smaller copies of itself.

Self similarity implies a kind of fairness when searching for an optimal design or result from an engineering equation. It means the same strategy is used whether you are looking at a design very similar to the current best one or very different to the current best one. Both strategies are merely scaled copies of each other.

How to Compute

You start out the design process with a randomly chosen 'parent' design. Then, using the self similar function you create a mutated 'child' design based on the parent. If the child turns out to be a better design than its parent then it simply replaces its parent, as the generator of new children. Over time and sometimes quite quickly, you can get very good designs.

An important difference with genetic algorithms is the type of mutation involved. Rather than being the small local mutations of genetic algorithms, the mutations can displace a particular design parameter from one extreme value to another with moderately high probability in one pass. On the other hand, it will often only make microscopic changes to other design parameters. This turns out to be a good search strategy, similar to the building blocks scheme used in sophisticated genetic algorithms.

Mutation Express

The easiest way to proceed is to standardize all the design parameters into the range from zero to one. For example, zero could mean 0 rpm for a motor and one could mean 100 rpm. Or, zero could mean a lever 1" long and one could mean a lever 5" long. To get a suitable mutation for each design parameter, you first pick a random number between zero and the precision number. You then calculate the exponential function of that random number. The exponential function produces a number somewhere between one and nearly zero. You then randomly decide whether to add or subtract this number to the design parameter to get the mutated version. If you

decide on a precision of -4, then you pick a number at random between zero and -4, for example -2.35. The exponential function of -2.35 is $0.0954 = \exp(-2.35)$. You then decide at random whether to add or subtract 0.0954 to the design parameter to get the mutated (child) version.

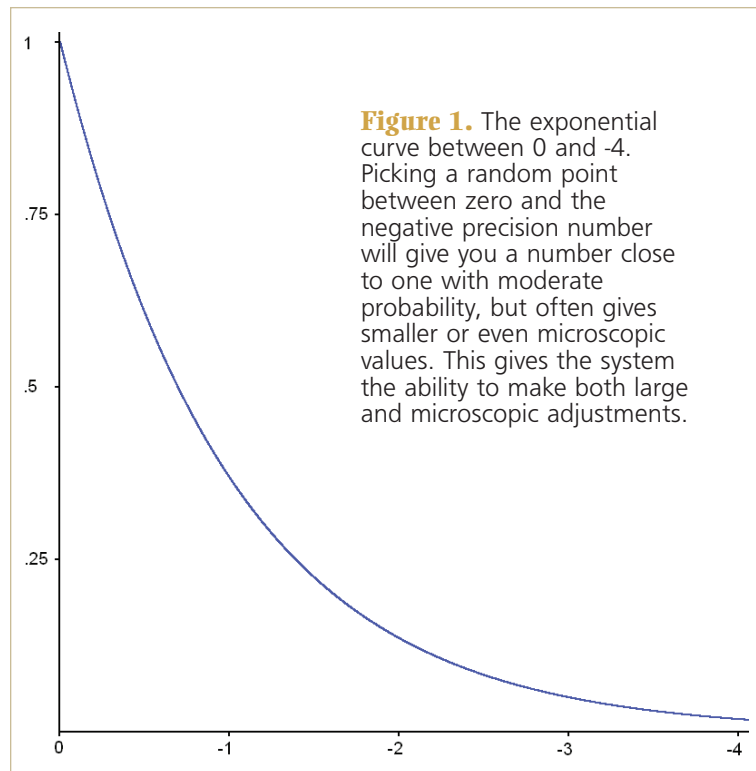
The only problem is that the design parameter can then become less than zero or more than one. Just clamp any numbers less than zero to zero, or more than one to one. **Figure 3** shows the system in action.

Performance and Applications

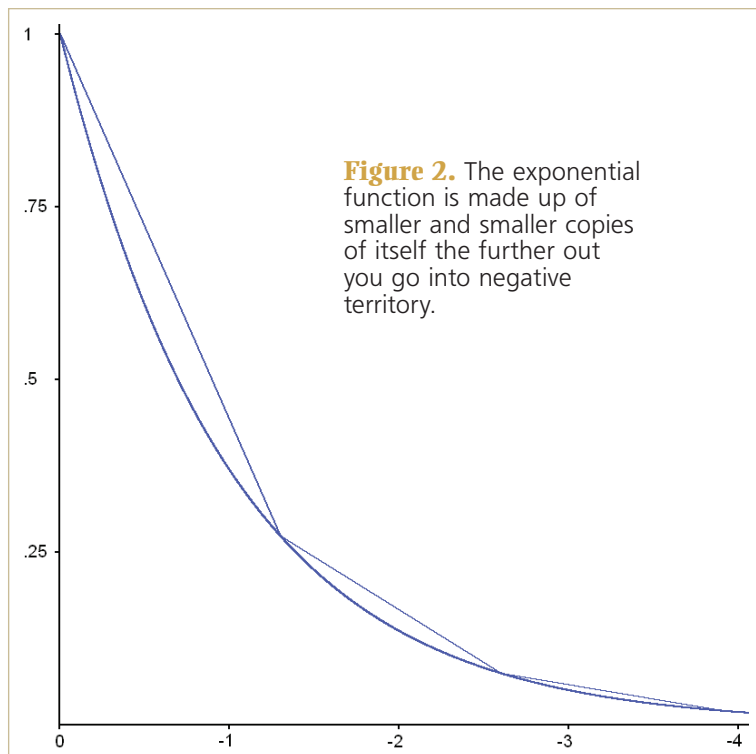
Experiments conducted with self similar optimization show that on many standard problems its performance can be similar to or better than standard genetic algorithms. Generally, such optimization techniques can give better results than a human designer in situations where the number of strongly interacting design parameters are too many for a person to cope with, and yet where the problem still retains some elements of simplicity about it. Unfortunately, it is also rather easy to pose intractable problems that neither human nor machine can solve. However, most engineering problems can be solved reliably. You can use it for so-called generative design (to design the shape of a building, lamp, or robot for example) or to create a robot brain using an evolvable neural net.

Precision

Most electronic components are available with a precision of 1% at best, 5% or 10% being more typical. Metal work can only be cut within certain tolerances. In the real world, you should not design a circuit where two resistors need to be equal to each other within .001%, for example. Genetic algorithms don't know any better and can easily come up with circuits like that. Self similar optimization



has a precision parameter that you can exploit to avoid unrealistic designs. For an electronic circuit design, you might choose a precision number somewhere between -5 and -25. If you are trying to design something more theoretical or complex (like a robot brain), then a precision number between -100 and -1,000 would be a good choice.



Changing Environments

Perhaps the most spectacular aspect of self similar optimization is its ability to continue to re-adapt to changing environments — something that genetic algorithms can hardly do.

For example, if you use it to design a neural net robot brain, the brain can learn to navigate around a particular room. If small changes in the room occur (such as a chair being moved), the self similar system can readily adapt to those changes. If the robot is suddenly put into a completely different room, that is not a problem either. It will start again — from scratch, if necessary — and re-adapt automatically to new situations without the need to press a reset button. Genetic algorithms in contrast converge to a particular solution. They cannot track large or even gradual changes. Changes in the environment really confuse them.

Co-Evolution

Another critical advantage to self similar optimization is its effectiveness for co-evolution. In nature, you can see co-evolution at work, like with plants and insects. An insect may adapt over the generations to eat the leaves of a particular plant. The plant may counter-adapt over the generations by developing thicker and less palatable leaves. This can lead to an arms-race between species, fueling the evolution of ever more complex forms. Co-evolution using self similar optimization is easier than any alternative. Each object in the environment has its own self similar optimizer, and continually appraises itself against the environment and tries to improve itself.

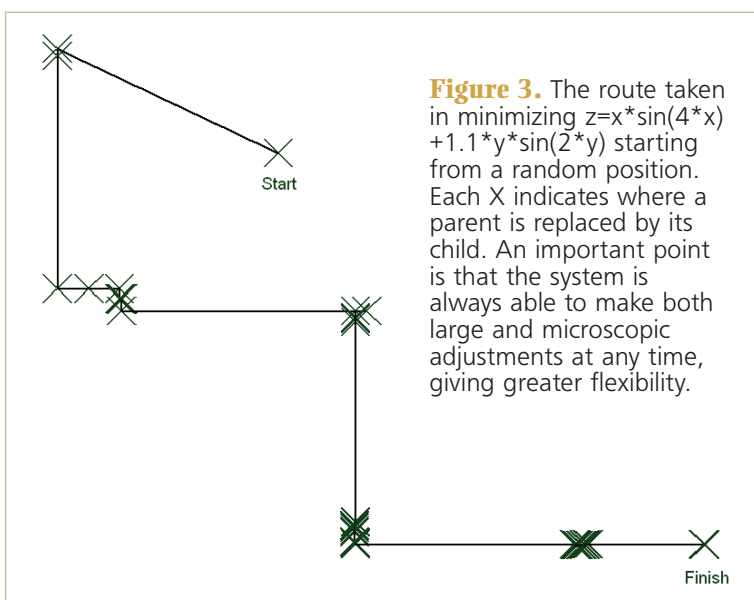
In a way, it is superior to what happens in nature, which uses a process similar to genetic algorithms that can lead to convergence and an inability to change. Many plants and animals have gone extinct because they could not adapt to radical changes in the environment, whereas if they had been equipped with a self similar optimizer they might still exist! There are some very simple examples of adaptive and co-evolving code to download and try at **Reference 1**.

Open Ended Evolution

Unending evolution with continuous improvement is a nice idea. It is true that sometimes genetic algorithms are run for days or even weeks, but you could end up wasting your time if they actually converge after only one day, for example. Self similar optimization at least allows for the theoretical possibility of open-ended evolution and the continuous accumulation of improvement. In reality, the rate of improvement does start to slow down after a while. Nevertheless, it is easy to create variations on the basic code — such as evolving the precision parameter along with the design parameters — that offer interesting possibilities for open-ended evolution.

Conclusion

An initial paper about self similar optimization or continuous gray code optimization (as it is sometimes known) appeared relatively recently (see **Reference 2**). It offers the robotics community a simple and useful tool. Even hobbyists who would avoid the complexity of coding a genetic algorithm can apply this idea and find ingenious uses for it, particularly for interactive robotics. It is not a miracle cure for every problem, but it is rich in potential applications. **SV**



References

1. <http://code.google.com/p/robotaware/downloads/list>
2. www.cs.bham.ac.uk/~jer/papers/ctsgray.pdf

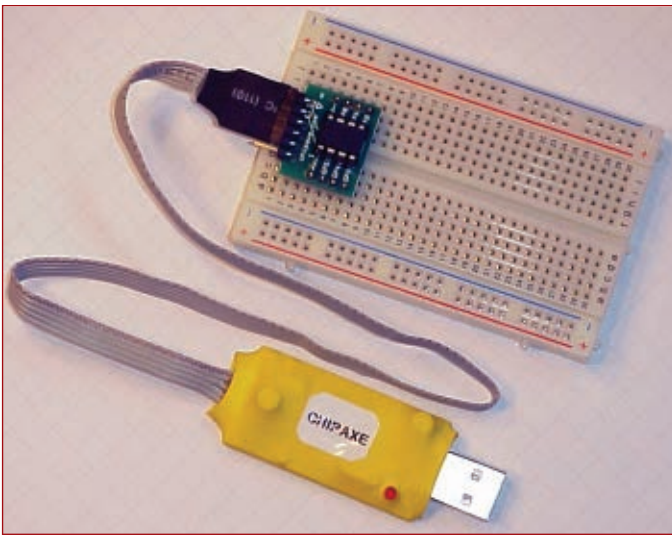


Figure 1. CHIPAXE Breadboard Starter Kit.

Simple Sensor Interface

When our CHIPAXE programming system was first introduced in the November *Nuts & Volts* "Getting Started with PICs" column, we were surprised by the great email response we received. Frankly, we were caught off guard by the large amount of interest. After all, CHIPAXE is just a simple concept conceived one night by a few friends during a round of beers. The original concept was to create an open source, low cost way to help people get started programming. What we found was you can't have a programming setup without offering various applications for someone to build, and none is more popular it seems than examples of how to work with sensors.

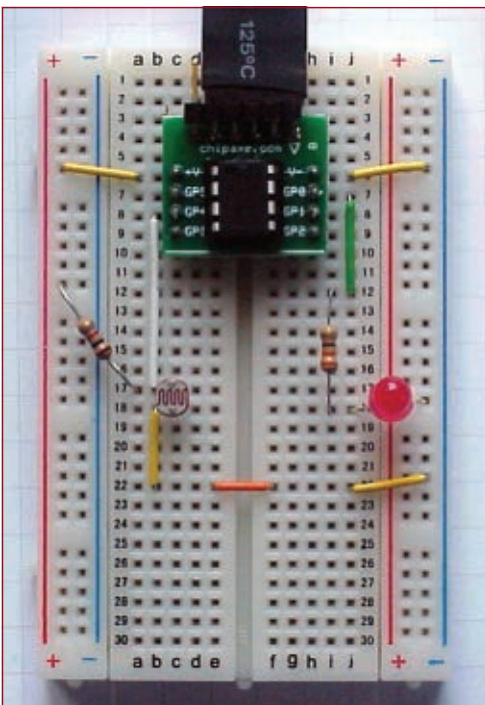


Figure 2. Final Light Sensor Project.

In this article, we'll attempt to describe a few extremely simple and low cost sensor applications that can be used for robotics or any electronic application. The logic board used for the projects is our own CHIPAXE Breadboard Starter Kit which has an eight-pin PIC12F683 microcontroller (**Figure 1**). The CHIPAXE system uses the sample version of the PICBASIC PRO compiler so the programming is simple for the beginner.

Project #1

In this project, we'll use the analog-to-digital converter (ADC) in the PIC12F683 to read a CDS cell which changes resistance as the light around it changes. A simple eight-bit resolution result will work perfectly. As the light changes, the ADC value will be tested. If it's a high value (high resistance), then it's dark out and a single LED lights up.

Figure 2 shows the final setup.

Hardware

The schematic is shown in **Figure 3**. We used a pull-up resistor in series with a light controlled resistor or CDS cell to form a resistor divider (or voltage divider) to convert the changing CDS resistance into a changing voltage. The pull-up resistor could easily be replaced by a potentiometer to give you a sensitivity adjustment. The output drives one LED on or off based on the outside light level (similar to a night light). The **Connection Table** describes the components and jumper connections on the CHIPAXE breadboard which has the rows and columns designated by numbers and letters, respectively.

Software

The start of the software requires some I/O setup since we will be using the GP4 pin as an analog input. Using the binary bit designation in PICBASIC PRO allows us to easily set the AN4 bit of the ANSEL register making it analog; the rest of the I/O pins are zero or set to digital mode.

```
ANSEL = %00001000    ' AN3/GP4 Analog,
                      ' GP2-GP0 Digital
```

The comparator on the 12F683 is shut down with this command line:

```
CMCON0 = 7 ' Comparator off
```

The state of the GP4 has to be set to input mode using the TRISIO register which can also set the rest of the I/O pins to outputs.

```
TRISIO = %00011000    ' GP4 input, GP2 thru
                      ' GP0 outputs
```

The ADCIN command requires some setup parameters to be established such as the ADC resolution, ADC clock source, and sampling time. These are easily done with DEFINE statements.

Software Listing 1

```
*****
'* Name      : CDS.BAS
*****

ANSEL = %00001000    ' AN3/GP4 Analog, GP2-GP0
                      ' Digital
CMCON0 = 7           ' Comparator off
TRISIO = %00011000    ' GP4 input, GP2 thru GP0
                      ' outputs

' Define ADCIN parameters
Define ADC_BITS 8     ' Set number of bits
                      ' in result
Define ADC_CLOCK 3    ' Set clock source
                      ' (3=rc)
Define ADC_SAMPLEUS 50 ' Set sampling time
                      ' in uS

adval var byte        ' Create adval variable to store
result

main:
ADCIN 3, adval        ' Read channel AN3 to adval

If adval > 150 then    ' Light LED if in the dark
High GPIO.0           ' Light all LEDs
ELSE
LOW GPIO.0
ENDIF

goto main             ' Loop Back to test light
                      ' sensor
```

```
' Define ADCIN parameters
Define ADC_BITS 8     ' Set number of bits
                      ' in result
Define ADC_CLOCK 3    ' Set clock source (3=rc)
Define ADC_SAMPLEUS 50 ' Set sampling time
                      ' in uS
```

A variable is established to store the ADCIN result:

```
adval var byte        ' Create adval variable to
                      ' store result
```

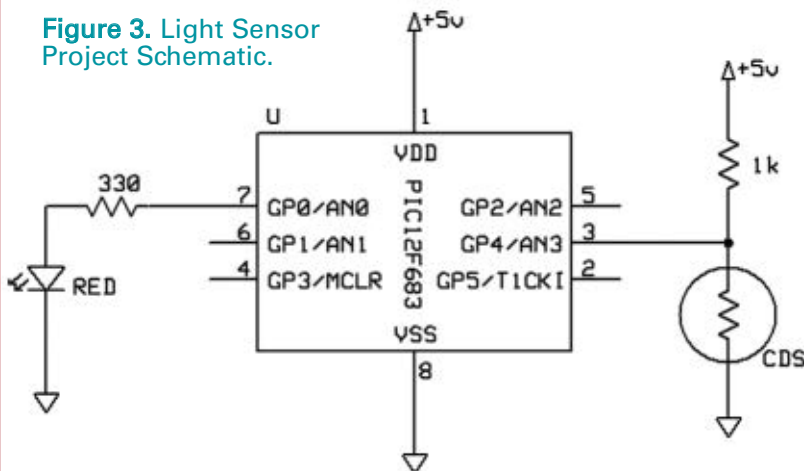
The main label establishes the main loop, followed by the ADCIN command line where the CDS cell is read.

```
main:
ADCIN 3, adval        ' Read channel AN3
                      ' to adval
```

After the value of the CDS cell is stored in the adval variable, it is compared to the value 150. If it is less than 150, then the LED is off. If the value is greater than 150, then it's dark and the LED is lit using the HIGH command.

```
If adval > 150 then    ' Light LED
                      ' if in the
High GPIO.0           ' dark
ELSE                  ' Light all
LOW GPIO.0            ' LEDs
ENDIF
```

Figure 3. Light Sensor Project Schematic.



Connection Table	
Micro	Pin 1 at C6
Yellow Jumper	a6 to +rail
Yellow Jumper	j6 to -rail
Green Jumper	j7 to j12
330 ohm	i12 to i18
Red LED	Anode j18, Cathode -rail
Yellow Jumper	j22 to -rail
Orange Jumper	f22 to e22
Yellow Jumper	b22 to b18
White Jumper	b8 to b17
1K ohm	a17 to +rail
CDS Cell	d17 to d18

Another GOTO statement completes the main loop.

```
goto main    'Loop Back to test potentiometer
```

Next Steps

Changing the threshold value from 150 to something higher or lower will determine how dark it has to be to light the LED. The limit is 0 to 255 since we used an eight-bit result. Just like the switch project, other sensors can replace the light sensor. A thermistor could replace the light sensor to measure temperature. A potentiometer could be used in place of the light sensor to create a mechanical interface. If you remove the pull-up resistor, then a Sharp GP2D12 object detection sensor that produces a variable output voltage can be read by the analog pin directly for more accurate robotic obstacle detection. As you can see, this simple setup can be used for many

sensor applications that are often required in robotics.

Project #2

This next project can be very useful for measuring impact or vibration of an object. The project uses a flexible piezo type vibration sensor that produces a voltage when vibration is sensed. By reading that voltage with the ADC inside the PIC12F683, the software can indicate

vibration is present by driving a piezo speaker to beep. This could be used as an alarm system to sense when an object is moved if the circuitry was attached to the item. It's sensitive enough to also detect foot steps. We've always wondered how much vibration a package would see when shipped through the post office. Maybe this will form the basis for that type of project in the future. The final setup is shown in **Figure 4**.

Hardware

The hardware consists of the CHIPAXE module positioned in the center of the breadboard with jumpers to the power rails so the programmer can power the whole circuit. The project uses a small vibration sensor from Measurement Specialties (part number 1005939-1) that produces a voltage when it's vibrating. The voltage output is fed directly into the GP4 pin which is set up in the software as an analog input the same way we did in the light sensor project.

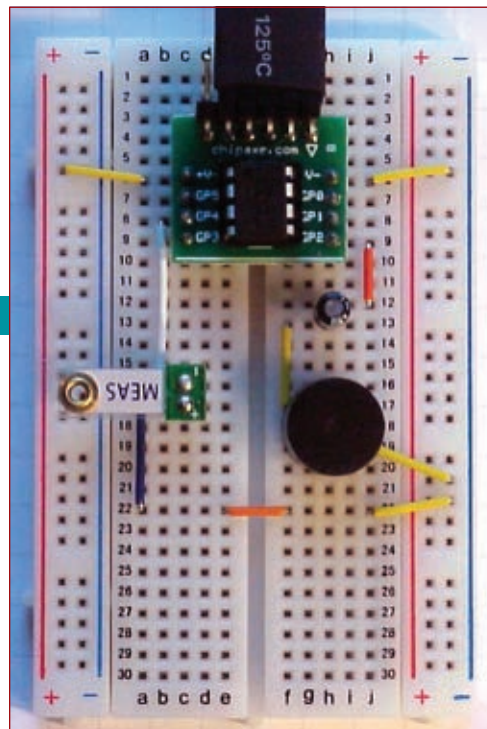
The GP2 pin is configured as a digital output that creates a square wave pulse using the software SOUND command to drive a piezo speaker. The square wave needs to be converted into a rounded signal to get the best possible sound with the piezo speaker so we place a 10 μ F capacitor in series between the GP2 pin and the piezo positive lead. The piezo speaker is then grounded. The piezo used is a DBX05-PN speaker from Jameco Electronics. The **Connection Table** describes how to connect all this on the breadboard and **Figure 5** shows the schematic.

One thing to note in the picture is the vibration sensor looks like it's in backwards with the positive pin connected to ground. This is not the case. The jumpers actually cross each other underneath the sensor to make the proper connection of negative to ground. Sometimes pictures don't tell the whole story but the **Connection Table** does.

Software

The software for this project is very similar to the previous one as it uses many of the same commands. We use the SOUND and the ADCIN command once again. To get started, we need to make the speaker pin digital except for the GP4 pin which will read the vibration sensor. We can set each bit by using the percent symbol in front of the 1s and 0s so the compiler knows we want

Figure 4. Final Sensing Vibration Project.



Connection Table	
CHIPAXE-8 Pin 1 at C6	
Yellow Jumper	a6 to +rail
Yellow Jumper	j6 to -rail
Orange Jumper	j9 to j12
10 μ F 35V	Positive-h12, Negative-h13
Speaker	Positive-g17, Negative-i19
Yellow Jumper	f13 to f17
Yellow Jumper	j19 to -rail
Yellow Jumper	j22 to -rail
Orange Jumper	f22 to e22
Blue Jumper	a22 to a16
White Jumper	b8 to b17
Vibration Sensor	Pos-d17, Neg-d16

to use a binary number to set each bit. In this case, all unused pins are also set up as digital pins.

```
ANSEL = %00001000 'I/O digital except AN3/GP4
Analog
```

The comparator is turned off by setting the CMCON0 register to seven.

```
CMCON0 = 7 ' Comparator off
```

The GP4 pin will sense the vibration sensor so we need to make sure it is an input by using the binary method to set all the pin states in the TRISIO register. A one makes a digital pin an input and a zero makes a digital pin an output. Even though we made GP4 an analog pin, we still have to make it an analog input. This seems redundant but is required inside the 12F683 microcontroller. An analog pin can be made into an output, but it won't work properly.

```
TRISIO = %00011000 ' GP4 input, GP2 thru GP0
                  ' outputs
```

The analog-to-digital settings for the ADCIN command are established with a set of DEFINE statements. These are the default settings so this step can be skipped. However, we like to define them in the code so we have the settings in writing. These do take up some of the 31 command limit, so if you run out of space with a program based on these lines, then you can comment them out, save command lines, and still know what the default settings are.

```
' Define ADCIN parameters
Define ADC_BITS 8 ' Set number of bits in
                  ' result
Define ADC_CLOCK 3 ' Set clock source (3=rc)
Define ADC_SAMPLEUS 50 ' Set sampling time
                      ' in uS
```

The ADCIN command needs to store the vibration value in a variable, so we have to create that variable. That is done in this line:

```
adval var byte ' Create adval to store result
```

The main loop of code starts with the familiar main: label.

```
main:
```

The ADCIN command is the first step and reads the GP4 pin which is also the AN3 pin or analog pin 3. The result is stored in the adval variable.

```
ADCIN 3, adval ' Read channel AN3 to adval
```

The next step is to test the value to see if vibration is present. This is where you set the sensitivity for the vibration sensor. The voltage of the vibration sensor will result in an ADC read of 0 to 255 since we are using eight-bit mode (set back in the Define command lines). If the voltage results in a value larger than 150, then a beep will be produced in the speaker. If you want to make it more sensitive, then lower this value. If you want to make it less sensitive, then raise the value.

```
If adval > 150 then 'Light LED if in the dark
```

If the vibration is large enough, then the SOUND command creates a square wave for a period of time. The

Software Listing 2

```
*****
'* Name : Vibration.bas
*****

ANSEL = %00001000 ' I/O digital except
AN3/GP4
                  ' Analog
CMCON0 = 7 ' Comparator off
TRISIO = %00011000 ' GP4 input, GP2 thru GP0
                  ' outputs

' Define ADCIN parameters
Define ADC_BITS 8 ' Set number of bits
                  ' in result
Define ADC_CLOCK 3 ' Set clock source
                  ' (3=rc)
Define ADC_SAMPLEUS 50 ' Set sampling time
                      ' in uS

adval var byte ' Create adval to
               ' store result

main:
ADCIN 3, adval ' Read channel AN3
               ' to adval

If adval > 150 then ' Light LED if in
                  ' the dark
sound GPIO.2,[100,100] ' Create sound for
                  ' 1/10 sec
ENDIF

goto main ' Loop Back to test
          ' vibration sensor
```

value is somewhere between those. The second value in the command line is the duration which can be between one and 255. Each value represents about a 12 millisecond period. In this case, the sound lasts about 1.2 seconds.

```
sound GPIO.2,[100,100]
  \Sound speaker for 1/10 sec
```

The If-Then command ends with the ENDIF command.

```
ENDIF
```

The GOTO command finishes the main loop of code by jumping the program control back to the main: label.

```
goto main      \ Loop Back to
                \ test the
                \ vibration sensor
```

Next Steps

This type of vibration sensing setup could be used in many applications where you need to test for movement. The sensor operates best in an up and down direction, so you might want to add a second sensor mounted at a 90 degree angle to the first one for another angle of sensing. Measurement Specialties makes the same sensor with a connector designed to mount at 90 degrees. That would be a great next step.

As we mentioned at the beginning, this setup could also be used to measure vibration on any item including something being shipped in a box to see how well the post office handles the package. To do that, you would need to store the value in memory and that requires an EEPROM (Electrically Erasable Programmable Read Only Memory) which is another topic for another article. (Besides, we don't think the post office will really allow a package with electronics running inside it to be shipped.)

Conclusion

We hope this article has helped you learn how to create a simple robotic or electronic sensor for your next gadget. If you'd like to build other simple projects like this with the CHIPAXE, feel free to stop by our website (CHIPAXE.com). Hope to see you the next time we get a chance to offer programming tips here in the pages of *SERVO*. **SV**

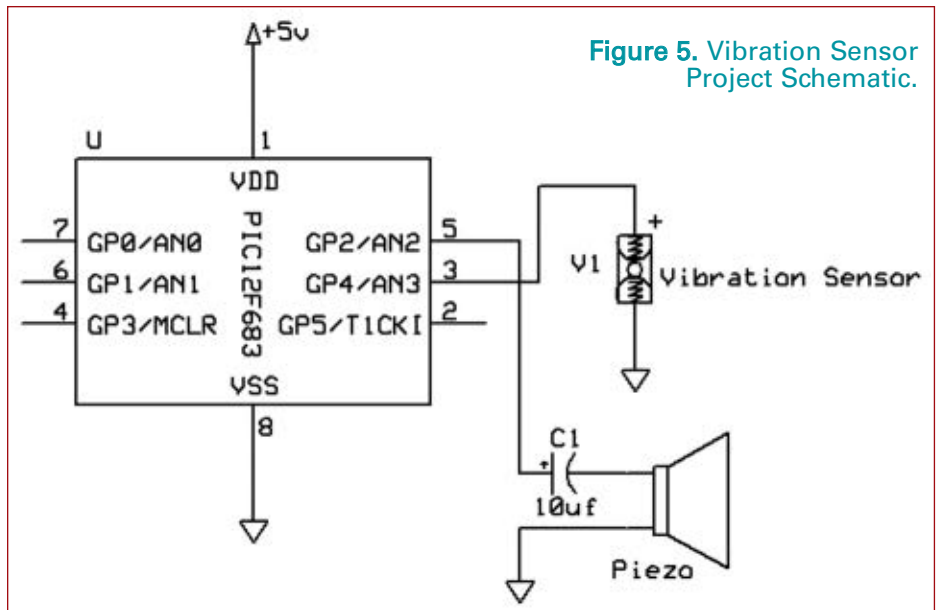



Figure 5. Vibration Sensor Project Schematic.



NOW IS THE TIME
To order your subscription to **SERVO Magazine**! Call us TODAY or visit our website to place your order.
877.525.2539 www.servomagazine.com

THE ORIGINAL SINCE 1994

PCB-POOL®

Beta LAYOUT

Servicing your complete PCB prototype needs :

- Low Cost - High Quality PCB Prototypes
- Easy online Ordering
- Full DRC included
- Lead-times from 24 hrs
- Optional Chemical Tin finish no extra cost

NEW ! FREE LASER STENCIL WITH ALL PROTOTYPE PCB-ORDERS

Watch "ur" PCB®
Follow the production of your PCB in **REALTIME**

email: sales@pcb-pool.com
Toll Free USA : 1 877 390 8541
www.pcb-pool.com

Beta LAYOUT

Hacker Dojos Offer Collaboration Opps for Robot Hobbyists

By Jeremy Kerfs

Developer communities help others, and collaborate for better projects. Making robots and gadgets by yourself can be great fun, but the enjoyment skyrockets when you can figure out thorny problems in collaboration with friends. For example, when the scripting language is beyond you, someone else can take over that aspect while you fix the gear box. This collaboration is not only used in hobbyist circles, but every well-run tech company encourages even the most brilliant engineers and scientists to work with each other, increasing productivity and creativity.

One group of hackers created a popular community in Silicon Valley to allow people to exchange ideas and hang out with others who shared their interest in engineering and developing. They started with a series of events called "Super Happy Dev House." These were hosted at corporate venues and sponsored by large tech companies. During the one day event, anyone could walk in and try out new inventions, talk to the creators, and make their own creations on the spot. Free Internet access, food, and fellow developers lured hundreds of people to the events.

Eventually, the founders decided that meeting occasionally was not enough, so they invested in a huge endeavor to create a permanent place for hackers and makers. The result was Hacker Dojo: a 4,400 foot building in Mountain View, CA. For \$100 a month, you are granted unlimited access to its facilities; you can come in anytime and tinker around the clock. Classes are held about subjects ranging from knot tying to programming Python and constructing circuit boards.

In the building, there is equipment ready for any project. Soldering irons and electronics components are put in a special room. In other rooms, couches alongside dozens of outlets are ready for programmers to start coding. A large room with projectors is set up for presentations and speakers.

Besides this normal everyday function, the facilities host

Random Hacks of Kindness Day



November
17th-20th

numerous events to encourage hackers to expand their skills, and produce meaningful gadgets and products. A special conference "Random Hacks of Kindness" came to Hacker Dojo on November 17th through 20th to provide a unique opportunity for the creative developers. The goal of the event was to design gadgets or programs that would help people survive or recover in the event of a natural disaster. At its heart, it's a competition designed to display a wide range of products created to solve many different aspects of a tragedy.

All of the contestants were judged based on three criteria: innovation, impact, and practicality. The event was sponsored by Microsoft, Google, Yahoo!, World Bank, and Nasa-Ames. Craig Fugate, a FEMA (Federal Emergency Management Agency) administrator, gave a keynote presentation detailing the importance of these types of projects. The requirements of the contest were generated based on the idea that most victims in a disaster have difficulty comprehending new and complicated technology, and can't use them effectively in a disaster situation.

Groups of contestants were formed and gathered together to discuss their plans. Some went for highly technical plans, while others believed that simple ideas would be the most practical. Each group had two days to create a prototype or a proof of concept to demonstrate their idea. With so little time, core functionality was essential; the finer aspects could be developed at later conferences or on their own time.

After time ran out, the contestants presented their ideas to the judges one at a time. The judges were all experts in their fields and had backgrounds with major technology companies. The short presentations had to prove that the idea was visionary and ready to be launched to every area that needed it; all met the constraints of easy to learn and use.

The Projects

Many innovative projects were presented by groups ranging from college students to seasoned professionals.

Two women from the University of Colorado started the presentation phase with their project "Tweak the Tweet." They focused on using Twitter to get useful and relevant information to emergency response crews from the survivors who witnessed the disaster first hand. It has been noticed during recent disasters that there may be hundreds of tweets sent containing information like "The house next door is on fire" or "I am heading north, away from the fire." This doesn't provide much information to rescue workers.

Instead, the college students suggested that a simple language could be set up by authorities to make the tweets easily searchable and readable. Examples included using "#addy" and then the address to specify where the people were located. Other notations would be about safe house location and how many people could fit there. With a

special search engine, this data could be easily accessible by emergency response personnel to help them quickly assess situations and determine appropriate measures.

The next group presented an SMS (Short Message Service) application that sends disaster information to people in the corresponding area. All a person has to do is type in his/her phone number and zip code into a website. Then, authorities could use the list and send messages to people in certain areas.

The key to this design is simplicity because it requires very little effort from the average citizen. It also provides a reliable way of spreading information if survivors could not get to TV or radios for updates on a disaster.

A third project focused on enabling citizens with the ability to provide unverified information that could help locate missing individuals. This project took the form of a website with a map and fields to fill in information about a missing person. The location, time, and description of the missing person would be captured. If this information could be corroborated by other individuals, the authorities would have a fairly good idea that the person was, in fact, alive and in that area.

Gathering together unverified information instead of only waiting for verified data from emergency crews could speed up the time it took to allocate resources to help the missing people. It also could provide the families of the missing person with more timely information on their loved one.

For another project, one team ("Break Glass") used the popular iPhone to construct an app that could easily and simply display the family's emergency plan. It also holds contact information for family friends that live out of the state. The advantage of this is the ability to find ways to meet up with and contact people even if the phone has no service or someone picked up the device and wanted to find out who it belonged to.

Requiring very little battery power, the app displayed what the person needed to do in order to locate family members and remain safe throughout the disaster. It also has the ability to quickly send out a message to everyone in the family with the push of a single button, including the updating of social networking sites such as Facebook and Myspace.

For another project, the group "I'm OK," created an app for iPhones that lets users easily send out a message to relatives — in the event of an emergency — stating they are okay. The simple interface and ease of contact made this convenient for victims in a disaster.

The application would alert any number of family members and did not require any further action from the user to send a simple message to others. This saved the user from having to send multiple text messages or calls.

Several other projects were presented. Each group offered unique solutions that would benefit people during a natural disaster. Some had creative ideas for using local

networks to restore connectivity in an emergency or process UAV (Unmanned Aerial Vehicle) imagery. FEMA awarded the team "Break Glass" (that created the family communication plan app) the FEMA prize, noting the accessibility of information and ability to contact others in multiple ways.

The Random Hacks of Kindness first prize was given to "I'm OK" as an excellent tool to alert others about their present condition. The students from the University of Colorado took second place with "Tweak the Tweet." Other groups got honorable mentions, and everyone was thanked for their contribution. The awards centered around communication problems because of the need for the reliable spread of information when traditional methods may not be reliable or accessible.

With so many unique and easily implementable ideas, Random Hacks of Kindness was definitely a success. Many engineers and developers enjoyed having the opportunity to hang out and talk about their favorite subjects, while they helped the future victims and survivors of natural disasters. This one conference was not the only event at Hacker Dojo that weekend, however.

Analog Synthesizers

In the next room, hackers were busy making music with their special analog synthesizers. They ranged from extremely complex, taking up a whole corner of the room to relatively small, almost handheld versions.

The goal of these inventions was to demonstrate sound creation with analog wires and circuitry instead of the digital products that are the mainstay of modern sound recording and playback. Along with the musical appeal comes the challenge and fun in making mechanical devices that control and operate the sound creation.

Seeing the inner workings of the machines in front of you instead of behind layers of plastic cases (like most devices today) showcased the ingenuity of the creators. There were approximately 20 unique devices illustrating many different ways of creating musical sounds.

Charles Merriam, the coordinator of the analog synthesizers event, was impressed by the quality of the machines produced. He asserted that "America's hobbies are better than most country's industries," highlighting the excellence of each synthesizer and how advanced it was.

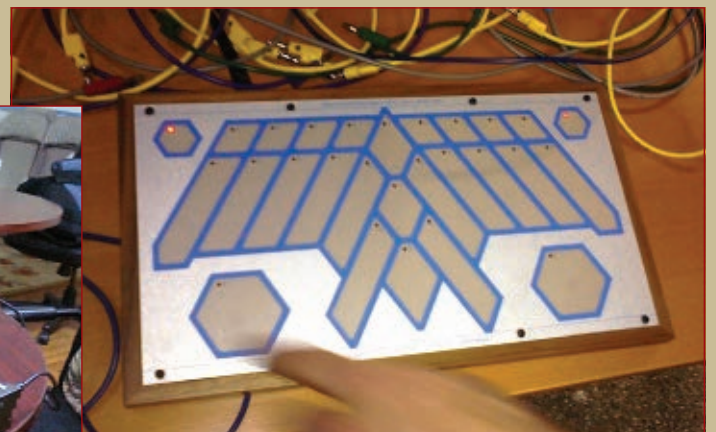
With these two adjacent rooms — both exhibiting the technical and creative talent of many individuals and groups — Hacker Dojo provided an excellent space for people of like-minds to congregate. One room emanated pulsating sounds while the other explored methods of saving people in disasters.

While these two events were unique, events just like them take place at Hacker Dojo frequently. The downtime between conferences is perhaps even more interesting, as members are able to use the space to hash out new ideas,

Making Music



Making More Music



Hacker Dojo has become well known in the San Francisco area, although many new, similar communities are growing throughout the country and world. Creating one is easier than ever with many people interested in working with others on every kind of project. Hacker Dojo grew from regular events that eventually reached more than 400 attendees. This method of getting together at specific times can be the starting point for a successful organization.

If enough people are interested and want to be able to spend more time together working on their projects, a dedicated building could be in order. Alternatively, corporate venues could house the larger events. Sponsorship by companies is also possible for developing clubs. When taking the step up to buying a facility, David Weekly (one of the founders of Hacker Dojo) warns to “never ever ever ask the city for a condition use permit for a dojo.” Besides that caveat, the rest of the process is different for every group.

Getting members for the organization — especially paying members — can be difficult. David advocates inviting colleagues and everyone that may be interested. Spreading the word via social networking sites is also a good idea. A slow expansion that builds the community is the perfect way of creating a dojo for developers and engineers to enjoy.

not restricted by the requirements of a specific conference or contest. Sometimes these plans can move on to real life products or remain an exciting hobby.

Closing Thoughts

Hacker Dojo brought all kinds of makers and hackers out of their offices and garages into an environment ready for developing the latest hobby technology advances. While Hacker Dojo is in Silicon Valley, other communities around the country and world have set up similar institutions.

Denmark is holding the next Random Hacks of Kindness conference as part of the Climate Change Conference. Super Happy Dev House — the event that inspired Hacker Dojo — has also expanded to diverse locations like Germany, the United Kingdom, and New Zealand. Besides large events, the number of hacker communities is also growing. **SV**

Complete Fabrication Center

AS 9100 Registration
In Process

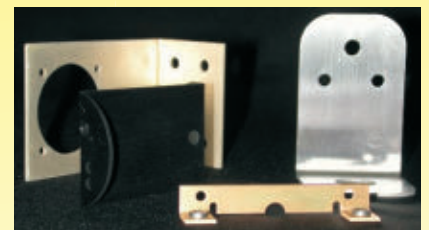
Quality prototype parts within 24 hours!!

**Precision Laser, Waterjet, Plasma, Machining,
Micro-Machining, Forming, and Welding Capabilities**

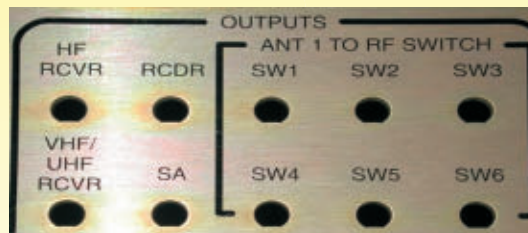


Parts from practically any material and
from 0.001" to 6.000" in thickness.

Finishes such as powder coat, paint,
plating, anodizing, silk screen, and
more!



**Fabricated, silkscreened and
shipped in 2 business days
with no expedite charges!**



- * **Quick**
- * **Affordable**
- * **Precise**
- * **No Minimums**

24 hour turn time based on quantity and finish requirements



Integrated Ideas & Technologies, Inc.

6164 W. Seltice Way • Post Falls, ID • 83854 • USA

Ph (208) 262-7200 • Fax (208) 262-7177 •

www.iitmetalfab.com



1st Annual Techno-Swap-Fest

Feb 13, 2010 - 9am to 2pm
National Electronics Museum
1745 West Nursery Road, Linthicum MD



Attention All Hobbyists

- *Is your workshop so cluttered that you can barely work on your projects?*
- **Did you buy something online that didn't fit into your plans?**
- ***Are you still looking for that perfect gadget, part or tool?***

If you answered YES or NO to any of these questions, you need to attend the 1st Annual **Techno-Swap-Fest** to buy, sell and gawk at all the precious junk that other hobbyists crave.

Here are just some of the hobbies expected to be represented:

- R/C (boats, planes, tanks, cars, trains, geese, monsters)
- Robotics (competition, combat, R2D2, micro-bots, junk-bots)
- Electronics (hackers, makers, crackers, phreaks, HAMs)
- Mechanical (steam engines, CNC, animatronics, whimsical)

And some of the precious stuff (a.k.a. junk) you'll find:

Motors, gears, tools, parts, radios, batteries, materials, wheels, circuits, gadgets, cases, plans, complete projects, never-completed projects, models, books, and much, much more



***** Door Prizes : Most Unusual, Least Useful, Best Dust Collector *****



All proceeds support the National Electronics Museum:

Entrance Fee:	Adults \$5, Kids under 16 FREE
Seller's Table:	\$10 (limited number, reserve early)
Club Sponsorship:	FREE (just send us your info)

For more information or to reserve a table send email to:

techno.swap.fest@gmail.com

<http://groups.google.com/group/techno-swap-fest>

*** Your choice for a wintry Saturday morning ***

- (a) Attend Techno-Swap-Fest and experience the thrill of hunting for junk
- (b) Help put up new wallpaper in the guest bathroom

The *SERVO* Webstore

Attention Subscribers ask about your discount on prices marked with an *

CD-ROM SPECIALS



5 CD-ROMs & Hat Special
Only \$ 109.00 (includes shipping)!

www.servomagazine.com



Pre-Order Special
Order now and we'll ship for FREE!
Estimated Ship Date- Mid January

ROBOTICS

PIC Robotics

by John Iovine

Here's everything the robotics hobbyist needs to harness the power of the PICMicro MCU!

In this heavily-illustrated resource, author John Iovine provides plans and complete parts lists for 11 easy-to-build robots each with a PICMicro "brain." The expertly written coverage of the PIC Basic Computer makes programming a snap – and lots of fun.

\$24.95



Forbidden LEGO

by Ulrik Pilegaard / Mike Dooley

Build the Models Your Parents Warned You Against.

Forbidden LEGO introduces you to the type of free-style building that LEGO's master builders do for fun in the back room. Using LEGO bricks in combination with common household materials (from rubber bands and glue to plastic spoons and ping-pong balls) along with some very unorthodox building techniques, you'll learn to create working models that LEGO would never endorse. **\$24.95**

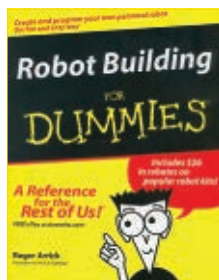


Robot Building for Dummies

by Roger Arrick / Nancy Stevenson

Discover what robots can do and how they work. Find out how to build your own robot and program it to perform tasks. Ready to enter the robot world? This book is your passport! It walks you through building your very own little metal assistant from a kit, dressing it up, giving it a brain, programming it to do things, even making it talk. Along the way, you'll gather some tidbits about robot history, enthusiasts' groups, and more.

\$24.95



Build Your Own Humanoid Robots

by Karl Williams
GREAT 'DROIDS, INDEED!

This unique guide to sophisticated robotics projects brings humanoid robot construction home to the hobbyist. Written by a well-known figure in the robotics community, *Build Your Own Humanoid Robots* provides step-by-step directions for six exciting projects, each costing less than \$300. Together, they form the essential ingredients for making your own humanoid robot. **\$24.95***



Robot Programmer's Bonanza

by John Blankenship, Samuel Mishal

The first hands-on programming guide for today's robot hobbyist!

Get ready to reach into your programming toolbox and control a robot like never before!

Robot Programmer's Bonanza is the one-stop guide for everyone from robot novices to advanced hobbyists who are ready to go beyond just building robots and start programming them to perform useful tasks.

\$29.95



Robotics Demystified

by Edwin Wise

YOU DON'T NEED ARTIFICIAL INTELLIGENCE TO LEARN ROBOTICS!

Now anyone with an interest in robotics can gain a deeper understanding – without formal training, unlimited time, or a genius IQ. In *Robotics Demystified*, expert robot builder and author Edwin Wise provides an effective and totally painless way to learn about the technologies used to build robots! **\$19.95**



We accept VISA, MC, AMEX, and DISCOVER
Prices do not include shipping and may be subject to change.

To order call 1-800-783-4624

SERVO Magazine Bundles



Published by T & L Publications, Inc.

\$57
per bundle

Save \$10
off the
normal
price!!

Now you can get one year's worth of all your favorite articles from *SERVO Magazine* in a convenient bundle of print copies. Available for years 04, 05, 06, 07, and 08.

Kickin' Bot

by Grant Imahara

Enter the arena of the metal gladiators!

Do you have what it takes to build a battle-ready robot? You do now! Here are the plans, step-by-step directions, and expert advice that will put you in competition — while you have a heck of a lot of fun getting there. Grant Imahara, the creator of the popular BattleBot Deadblow, shares everything he's learned about robot design, tools, and techniques for metal working and the parts you need and where to get them.

\$24.95

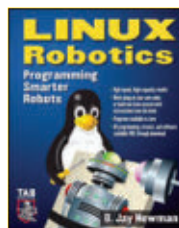


Linux Robotics

by D. Jay Newman

If you want your robot to have more brains than microcontrollers can deliver — if you want a truly intelligent, high-capability robot — everything you need is right here. *Linux Robotics* gives you step-by-step directions for "Zeppo," a super-smart, single-board-powered robot that can be built by any hobbyist. You also get complete instructions for incorporating Linux single boards into your own unique robotic designs. No programming experience is required. This book includes access to all the downloadable programs you need.

\$34.95



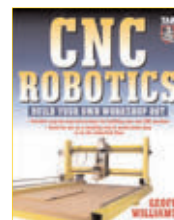
CNC Robotics

by Geoff Williams

Here's the FIRST book to offer step-by-step guidelines that walk the reader through the entire process a building a CNC (Computer Numerical Control) machine from start to finish. Using inexpensive, off-the-shelf

parts, readers can build CNC machines with true industrial shop applications such as machining, routing, and cutting—at a fraction of what it would cost to purchase one. Great for anyone who wants to automate a task in their home shop or small business

\$34.95



Visit my online store @
www.servomagazine.com

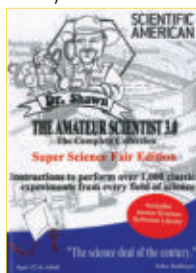
SPECIAL OFFERS

The Amateur Scientist 3.0 The Complete Collection

by Bright Science, LLC

There are 1,000 projects on this CD, not to mention the additional technical info and bonus features. It doesn't matter if you're a complete novice looking to do their first science fair project or a super tech-head gadget freak; there are enough projects on the single CD-ROM to keep you and 50 of your friends busy for a lifetime!

Reg \$26.95 Sale Price \$23.95



Getting Started Combo



The Getting Started Combo includes: *Getting Started in Electronics* by Author Forrest Mims and the DIY Electronics Kit. In his book, Mims teaches you the basics and takes you on a tour of analog and digital components. He explains how they work and shows you how they can be combined for various applications. The DIY Electronics Kit allows for the hands-on experience of putting circuits together — the kit has over 130 parts! No soldering is required and it includes its own 32 page illustrated manual.

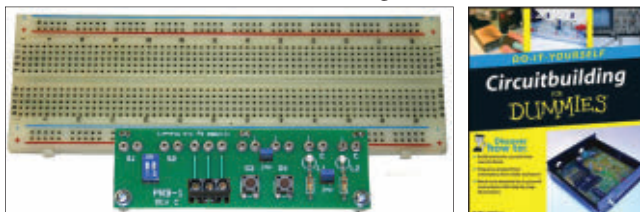
Combo Price \$62.95 Plus S/H

SPECIAL OFFERS

Proto Buddy Kit & Book Combo

For those just getting started in electronics as a hobby, a solderless breadboard (SBB) is the perfect platform for building those first circuits. Attach a Proto Buddy to an SBB, include a battery or two, and you will have a combo that has a lot of the same functionalities as more expensive units.

Combo includes PCB & Components, 830 point SBB, and Do-It-Yourself Circuitbuilding For Dummies.

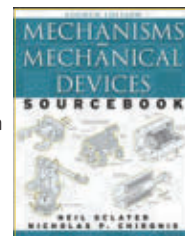


Combo Price \$57.95 Plus S/H
Limited time offer.

Mechanisms and Mechanical Devices Sourcebook

by Neil Sclater,
Nicholas Chironis

Over 2,000 drawings make this sourcebook a gold mine of information for learning and innovating in mechanical design. Overviews of robotics, rapid prototyping, MEMS, and nanotechnology will get you up-to-speed on these cutting-edge technologies. Easy-to-read tutorial chapters on the basics of mechanisms and motion control will introduce those subjects to you. **Reg \$89.95 Sale Price \$69.95**



Getting Started With PICs Volume 2

by Chuck Hellebuyck

Getting Started With PICs Vol 2 continues where the first book left off. This book contains the collection of the next 12 articles that were published in *Nuts & Volts* in 2007. A great compliment to Vol 1 and a must have if you're entering the world of PICs and programming.



Reg \$29.95
Introductory Price \$24.95

Enter into the world of
PICs & Programming
with this great
combo!

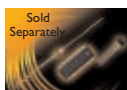
Combo Price \$155.00

For complete details visit our webstore @ www.servomagazine.com

PROJECTS

Das Blinkenboard Kit

As seen in *Nuts & Volts* June issue, Personal Robotics. by Vern Graner
This kit includes a pre-programmed ATtiny84 microcontroller that sports eight software PWM channels to control motor speed and light brightness. Jumper selectable patterns can be used to operate motors, solenoid valves, relays, or any DC load up to 24V/500 mA per channel!
Expand your board with the "Der Magnetfelder Detektor" componet pack.



Subscriber's Price \$32.45
Non-Subscriber's Price \$35.95
PCBs can be bought separately.

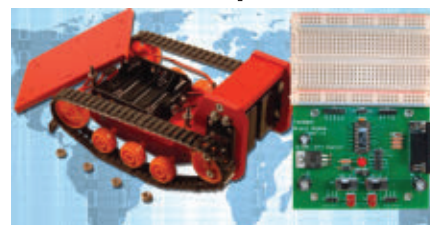
16-Bit Micro Experimenter Board



Ready to move on from eight-bit to 16-bit microcontrollers? Well, you're in luck! In the December 2009 *Nuts & Volts* issue, we introduced the new "16-bit Micro Experimenter." The kit comes with a CD-ROM that contains details on assembly, operation, as well as an assortment of ready-made applications. New applications will be added in coming months.

Subscriber's Price \$55.95
Non-Subscriber's Price \$59.95

Tankbot Kit & Brain Alpha Kit



As seen in the Sept. issue
Tankbot/ Brain Alpha
by Ron Hackett

A series filled with projects and experiments to challenge you through your learning process while you grow your fully expandable Brain Alpha PCB!

The brain is a PICAXE-14A!

For more info & pictures, visit the *SERVO* Webstore.
Tankbot and the *Brain Alpha Kit* can be purchased separately.

Combo Price \$ 138.95

EVENTS

Calendar

ROBOTS.NET

Send updates, new listings, corrections, complaints, and suggestions to: steve@ncc.com or FAX 972-404-0269

Know of any robot competitions I've missed? Is your local school or robot group planning a contest? Send an email to steve@ncc.com and tell me about it. Be sure to include the date and location of your contest. If you have a website with contest info, send along the URL as well, so we can tell everyone else about it.

For last-minute updates and changes, you can always find the most recent version of the Robot Competition FAQ at Robots.net: <http://robots.net/rcfaq.html>

— R. Steven Rainwater

JANUARY 2010

- 20-23** **Kurukshetra**
Guindy, Chennai, India
Events include Designer's Quest, Solenopsis, Freightcog, and Catch Me If You Can.
www.kurukshetra.org.in
- 22-24** **Techfest**
Indian Institute of Technology, Bombay, India
Events include Micromouse, Pixel, Full Throttle: Grand Prix, and Blitzkrieg.
www.techfest.org
- 26-28** **Singapore Robotic Games**
Singapore Science Center, Republic of Singapore
Lots of events including Micromouse, Sumo, robot soccer, wall climbing, pole balancing, underwater robots, legged robot marathon, legged robot obstacle race, "robot colony," humanoid competition, and more.
<http://guppy.mpe.nus.edu.sg/srg>
- 28-31** **Robotix**
IIT Khargpur, West Bengal, India
Apparently all the events at Robotix must have an X in their name: Xants, TribotX, Xtension, 8mileX, Xplode, and eXplore.
www.robotix.in

FEBRUARY

- 18-21** **Teckkriti RoboGames**
Indian Institute of Technology, Kanpur, Uttar Pradesh, India
Events include automated car parking, robot

combat, and robot ball games.
www.techkriti.org/#/competitions-robogames

- 21-25** **APEC Micromouse Contest**
Washington, DC
Very fast, very small robots solve mazes very quickly.
www.apec-conf.org
- 25-28** **Pragyan**
National Institute of Technology, Trichy, India
Micromouse, square route line following, and slam dunk robot basketball.
www.pragyan.org

MARCH

- 6** **CIRC Central Illinois Bot Brawl**
Lakeview Museum, Peoria, IL
Events include RC combat, miscellaneous Sumo, line following, line maze, and Best in Show.
<http://circ.mtco.com>
- 12-13** **AMD Jerry Sanders Creative Design Contest**
University of Illinois at Urbana-Champaign, IL
Robots must move balloons around a field to play Tic-Tac-Toe. To make it more challenging, teams must perform specific tasks in order to obtain their balloons.
<http://dc.cen.uiuc.edu/>
- 20-21** **Manitoba Robot Games**
TecVoc High School, Winnipeg, Manitoba, Canada
Events include Sumo, mini Sumo, line following, robot tractor pull, SuperScramble, Frequent Flyer, COGMATION, and Robo-Critters.
www.scmb.mb.ca
- 20-21** **RobotChallenge**
Vienna, Austria
Events included in this competition are parallel slalom, slalom enhanced, standard Sumo, mini Sumo, and micro Sumo.
www.robotchallenge.at
- 20-21** **Roboticon**
University of Guelph, Ontario, Canada
Robotic soccer for LEGO robots.
www.collegeroyal.uoguelph.ca

Twin Tweaks

THIS MONTH:

The Education of Budding Roboticists



COSMOS CLUSTER 2.

In previous installments of Twin Tweaks, Evan has had the privilege of chronicling his education as a mechanical engineering major at UCSD, because it just so happens that robotics are an excellent way to prepare engineering students for real world problems that demand interdisciplinary knowledge, teamwork, and technical communication skills. Robotics projects have been a critical part of Evan's studies as a mechanical engineer, from the ring stacking robot contest of MAE 3, the lower division design course (August '06 Issue), to the sophisticated safe cracker of MAE 156B, a senior design course (January '09 Issue). But while *SERVO* readers may now be convinced that engineering students are given a generous dose of robotics during the course of their education, there is yet another important issue where a careful application of robotics can be the solution. The adequate preparation of engineering students in American schools is not really at issue, but the numbers of students pursuing those degrees are. Numerous organizations — perhaps most notably FIRST — have recognized that robotics are an effective and inspirational way to both prepare high school students for the rigors of an engineering program and to inspire them to pursue it in the first place. During the summer of 2009, I was given an inside look at how the COSMOS program seeks to achieve those same results.

A COSMOS That Carl Sagan Would Be Proud Of

COSMOS is the California State Summer School for Mathematics and Science — a program that offers a variety of four week residential programs hosted at several campuses of the University of California system. The concentrations



THE CLUSTER 2 TEAM, (FROM TOP): DR. NATE DELSON, DR. RAYMOND DE CALLAFON, BRINN BELYEA, EVAN WOOLLEY, AND ZAC DOOLEY.

of the programs (or “clusters” in COSMOS lingo) range from biomedical engineering to oceanography to robotics. The participants are high school students entering the ninth through the 12th grades. The tuition for the summer 2009 program was \$2,550 for California residents and \$6,500 for non-residents. That may sound steep, but the program offers a lot. The tuition pays for on-campus residence (a great introduction to college living for the students, who even have resident advisors) and food at the dining halls for the four weeks (another essential introduction to college life). But tuition pays for much more than the bare necessities. Transportation for field trips is also provided, but the best part of all of this is the attention that the students get from real university faculty members. The cluster that I was involved with was staffed by two full time professors (the incomparable Dr. Nathan Delson and Dr. Raymond de Callafon), a high school physics teacher (the fantastic Mr. Brinn Belyea of Torrey Pines High School), and two recent graduates from UCSD, myself and fellow mechanical engineer Zac Dooley. All of this brainpower was brought together to inspire a group of 23 high school students in Cluster 2 about the joys, trials, and tribulations of building a mechanical clock and a kinetic sculpture.

The folks at COSMOS are dedicated to keeping this as an accessible program, because students can apply for financial aid to pursue this great opportunity. Approximately 40% of the students for the 2009 COSMOS program received full or partial financial aid.

COSMOS-politan

Evan wanted to be involved with the COSMOS program because he remembered how inspira-

tional the engineering programs we went through in high school were. During three weeks at the Palo Alto Research Center, we had the opportunity to see the types of real world projects that engineers worked on, and the FIRST program gave us a dose of intense interdisciplinary teamwork where we experienced the sense of accomplishment that comes from completing an intimidating project. These were the experiences that inspired Evan to go into engineering — an admittedly daunting field — because completing these programs comes with the realization that with the right team and the right drive, anything is possible.

The other Cluster assistant, Zac Dooley, was similarly dedicated to inspiring students about engineering, and he had served as an MAE 3 undergraduate tutor like Evan had. That means they both already had the opportunity to work for Dr. Delson. *SERVO* readers may remember Dr. Delson as the mastermind behind both the MAE 3 (ring-stacking robot) and MAE 156A (safe-cracking robot) projects. Joining Dr. Delson was Dr. de Callafon, an expert on controls that would be teaching the COSMOS students the ins and outs of programming in Basic. To complete the team was Mr. Belyea who would teach the students physics fundamentals that would be essential in building their kinetic sculptures. Kinetic sculptures and programming, you ask? That’s not even the half of it!

If You Want To Start From Scratch, Start With The Creation Of The Universe

The COSMOS program is a seemingly short four weeks, but in Cluster 2 we hit the ground running. After some quick introductions, the

CLUSTER 2 HARD AT WORK ON THEIR CLOCKS.



EVAN HELPS OUT AARON AND AMARAJ WITH THE LASERCAMM.





ZAC HELPS OUT ANDREW AND DANEE WITH SOME PROGRAMMING.



VISITING THE STINGRAY.



THE COSMOS CONTROL BOX.

students were thrown head first into their first project: a mechanical clock project eerily similar to the one presented to mechanical engineering students in MAE 3 at UCSD. The COSMOS students were expected to design a pendulum and escapement wheel on AutoCAD, and perform a timing analysis using the simulation program Working Model. A tutorial on AutoCAD was on the menu for the very first day. By the end of our first session, the students had already designed fierce looking escapement wheels.

The mechanical clock project is a great introduction to many fundamentals that are critical for any engineering project. The swinging pendulum is wonderfully illustrative of conversion from kinetic energy to potential energy, center of mass, and moments. Fabricating the clock is also a great introduction to basic shop tools and the super cool Lasercamm for rapid prototyping. The clock project is a fabulous way to practice the most essential skill in any engineer's toolbox: creativity. Students were able to exercise their creativity by designing their own pendulums. Pendulum designs ranged

from Atlas holding the world, to microphones to dragons and everything in between.

You Must Be Kessler

The first week was spent wrapping up the mechanical clocks, so that in the second week we could get started on the main event: kinetic sculptures. Kinetic sculptures, you ask? This is a robotics magazine, not an art magazine! But these kinetic sculptures are automated, with a Basic ATOM as a brain and servo motors for locomotion.

The kinetic sculpture project is a group activity where teams of 3-4 students can quite literally let their imaginations run wild. The sculpture is where the Cluster curriculum diverges from the MAE 3 syllabus, with the sculpture replacing the robot contest. Taken as a whole, however, the kinetic sculpture project is quite similar to the robot contest. The requirements of the sculpture are that it must have several mechanisms that respond to both sensor input and manual control — much like a robot. The competitive aspect comes in when

each team tries to make their sculpture bigger, better, and cooler than the next team's.

The main brain of the kinetic sculpture is the COSMOS controller box. The controller box uses a Basic ATOM microcontroller, an LCD screen, and inputs and outputs for the sensors and actuators. The sensors used with the sculptures are optical speed sensors containing two sets of phototransistors. The phototransistors can count the number of passing balls (the bright yellow passengers of the kinetic sculpture thrill ride) or calculate their speed based on the distance between the phototransistors and the time of each reading. Traditional limited rotation servomotors were used for the actuators.

The skeleton of the kinetic sculp-



CASINO DE GEISEL BY THE JUSTICE LEAGUE.

Twin Tweaks ...

ture is an intriguing set called the Chaos Tower, which is meant to be a fun and interactive introduction to physics. The Chaos Tower includes a basic structure of blue cylindrical beams, plastic connectors, and a track for some intrepid yellow balls. The kit includes a multitude of ways to see physics in action, including trampolines, to illustrate the coefficient of restitution and vertical loops to demonstrate centripetal force. The Chaos Tower forms the foundation for the COSMOS kinetic sculptures, and the students are given the task of automating the sculptures.

Two examples that we provided to kick-start their imaginations were a moving trampoline and moving basket. The trampoline used a random number generator to change positions to bounce a ball into one of three baskets. The moving basket used data from a speed sensor to change its position to catch balls approaching at different speeds.

To supplement the basic kit, we provided the students with the speed sensors, servo motors, potentiometers, switches, and generous amounts of acrylic for use with the Lasercamm.

These Pretzels Are Making Me Thirsty

For the purposes of the kinetic sculpture project, the students of Cluster 2 were divided into five teams of four and one team of three. For Week 2, students were tasked with building a mini-sculpture that used at least one sensor and one actuator. To aid in the designing and building of the mini-sculptures, students were given instruc-

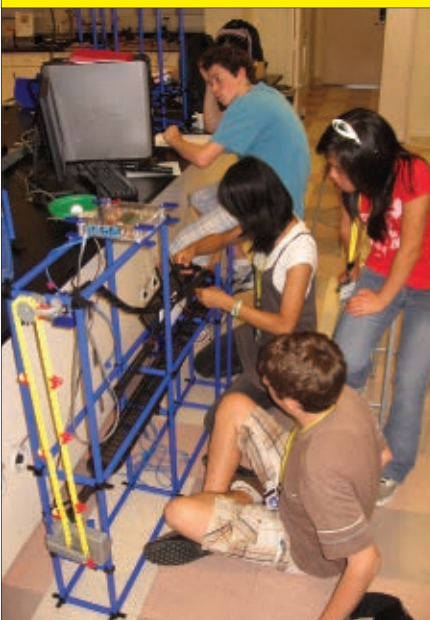


CIRQUE DU GEISEL BY THE BOYZ IN DA HOOD.

tion in physics by Brinn and programming by Dr. de Callafon. But as any good project manager knows, the brainstorming process can be a tumultuous and unruly tempest, so Dr. Delson sought to provide some guidance with teamwork and prototyping exercises. Teamwork was solidified with the classic Oreo cookie tower exercise (building the highest tower possible out of Oreo cookies); brainstorming required drawing ideas out on paper and risk reduction activities.

Risk reduction exercises were essential for the COSMOS program because the compressed schedule forced students to make quick design decisions. To test the feasibility of their designs, the teams were equipped with foam core boards and a working knowledge of the Lasercamm. Many ideas for basket mounts and servo arms proved fruitful, but perhaps the best discoveries were of

BLUE STEEL'S LEARNING SCULPTURE.



TEAM MAGNUM'S DIVERGING PATHS.



A STEEP SLALOM AND GRADUAL SLIDE BY WE THE QUEENS.



the designs that needed some work. How could jamming be avoided on the linear slide? What would be a more optimal gear ratio? Finding what doesn't work or what could be done better is truly what motivates good design.

In Cluster 2, we aimed to give the students a comprehensive exposure to the world of engineering and robotics, and many other activities provided respites from kinetic sculpting. At a set of outdoor physics activities, we shot bottle rockets, fired water guns, sling shot water balloons, and dropped bouncing balls off of the engineering building. While this might all seem like fun and games, the inclusion of inclinometers, coefficients of restitution, and cosines revealed these activities to be essential reinforcements of engineering fundamentals. Because no matter how advanced the project or how independent the robot, the fundamentals are always there.

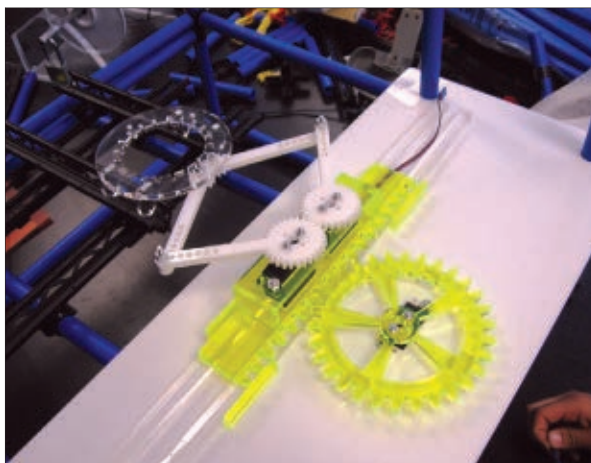
Speaking of advanced robots, one of our vicissitudes was a visit to see the Stingray — UCSD's entry for the AUVSI competition. We covered the AUVSI event in the August '06 issue of *SERVO*. It challenges teams to build an autonomous underwater robot capable of completing tasks like pipe inspection and buoy release. The great thing about visiting the Stingray was that it showed the COSMOS students the power of engineering; they got to see how tough and usually dry subjects like programming can come to life in an exciting way. It was just the kick-start they needed to wrap up their kinetic sculptures.

Made Of Star Stuff

At the end of our furious four weeks, all of the COSMOS students had put together some amazing sculptures; all with unique themes but united in the fact that they all presented a commitment to and excitement about the world of robotics and engineering.

Team 1 was the Justice League, comprised of Danee, Amara, Robert, and Andrew. They took a gamble when they created the Casino de Geisel, complete with a roulette wheel, pachinko machine, and moving basketball hoop. One of their biggest challenges was perfecting the basketball hoop which moved vertically up and down to catch the balls. It was a perfect linear slide problem, and the Justice League saved the day by customizing some long bearings and by rigging up their lift to apply the upward force as evenly as possible.

Team 2 was the Boyz In Da Hood and included Trevor, Jonah, Amaraj, and Jonathan. They invited us to the death-defying Cirque Du Geisel (Dr. Seuss would be proud). This circus included heart-stopping jumps powered by two

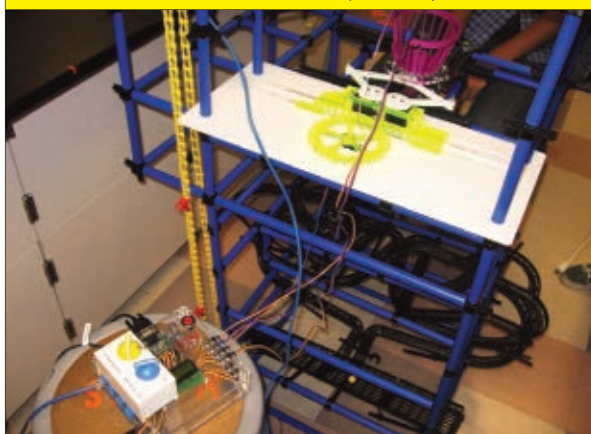


THE CUSTOM MECHANISM FOR TEAM 6'S CATCHING BASKET.

non-geared motors. The Boyz had tested out the non-geared motors in their mini-sculpture, and to do so they used the MotoMaster Motor Controller normally reserved for the MAE 156A students. They were even able to work out a program that allowed them to randomize the speed of the motors, just in case the jumps were not yet death-defying enough.

Team 3 was Blue Steel, made up of Amaris, Christine, Aaron, and Rafael. They took on the incredibly ambitious task of building a sculpture that could "learn." Blue Steel's build was truly a compelling case study on optimization, which is certainly an important concept in any field of engineering and the focus of the MAE 156A safe-cracking robot. The basic structure of their sculpture was deceptively simple — a nice slope comprised of flexible track pieces. The slope took a hairpin turn at the end, which then funneled balls back to the carrier. A servomotor was situated at the bottom of the slope with an arm that could pull back and forth so that it could increase or decrease the tension in the track and change its

THE SCULPTURE FROM THE TEAM TOO NINJA FOR A NAME (TEAM 6).



slope accordingly. The track was designed so that when pushed to its steepest, the ball would fly off the hairpin turn and never make it back to the ball carrier. If the track was pulled to its most gradual slope, the ball would complete its journey, but at a slow pace. Blue Steel's self-imposed challenge was to use feedback from speed sensors and ball counters to optimize the track – make it so the ball could complete its journey as quickly as possible without falling off. In four weeks, they were able to complete this gargantuan task, and they had a sculpture with programming so sophisticated it might even give the Stingray a run for its money.

Team 4 was Magnum, consisting of Eric, Javier, Daniel, and Cesa. Their sculpture included diverging paths and a harrowing labyrinth. The paths diverged by a ball tray that could change position using a servomotor controlled by a button. The guys on Team Magnum also put their newly-found AutoCAD skills to work by cutting out a team logo on the Lasercamm.

Team 5 was We The Queens, with Rocio, Carla, and Joana. Their sculpture featured a steep slalom with a catching basket. Their sculpture also included diverging paths; one was designed to achieve high speed and the other took a more languid pace. The two paths converged before a catching basket; the Queens aimed to create as big a difference in ball speeds as possible to really test the reaction of the catching basket. Despite some trying times with the speed sensors and the programming, the Queens completed their sculpture, and they successfully showcased a catching basket with a huge range of motion.

Team 6 – comprised of Stephanie, Jennifer, Su, and John – was simply Team 6, because they were too ninja for a name. Their sculpture showcased a super cool catching basket that was manually controllable using a custom control box. The ninjas of Team 6 wanted to create a catching basket that could move in both the x and y directions. They cut out a mechanism that was a combination of a rack and pinion and a scissor arm using fluo-

rescent green acrylic that matched the coolness factor of the mechanism. To control the mechanism, the ninjas created their own control box using two potentiometers with knobs as controls. Moving trampolines created the uncertainty that would test the ninja reflexes of anyone at the controls, but the responsive mechanism allowed even the non-ninjas among us to save the yellow ball from off-track oblivion.

All That Is Or Ever Was Or Ever Will Be

The robotic sculptures were all a success, but a real engineering project involves so much more than the project itself – science is not performed in a vacuum! In addition to building the sculptures themselves, the COSMOS students also had to construct a website and prepare an oral presentation. The designs of the sculptures themselves needed to be supported by a plethora of analysis, including physics calculations and mathematical simulations. We were even able to have some fun with a high speed camera, which was ostensibly an analytical tool that allowed us to verify the results of the simulations (even though it really made us feel like we were on Mythbusters).

The websites and presentations challenged the students to hone their communication skills, because the greatest engineering solution in the world isn't great at all if you can't explain it. One of the best parts of the presentations was to see the genuine sense of pride that everyone radiated. Building a robotic sculpture with servos and sensors and programming may have seemed like an insurmountable task at the beginning, but in four weeks they built one. Plus they built a clock and designed a website!

It was undeniably a challenging and fulfilling time for everyone involved. After a barrage of lectures and lab time, all of the students had created something to be proud of. Perhaps the best thing that everyone gleaned from COSMOS was confidence. Robotics, science, and engineering are all daunting fields which many students may avoid in college because they can't imagine themselves being an engineer and solving the tough problems day in and day out. Cluster 2 posed a tough problem, and every single student came up with a creative solution that showed they really can be engineers.

If this sounds like a great opportunity for you or someone you know, check out more about COSMOS online and consider submitting an application. The application period for the 2010 program is February 1st through March 15th. Who knows ... it might just change your life. **SV**

RECOMMENDED WEBSITES

<http://maelabs.ucsd.edu/cosmos/>
www.jacobsschool.ucsd.edu/cosmos/

SPECIAL THANKS TO

Dr. Nathan Delson
Dr. Raymond de Callafon
Brinn Belyea
Zac Dooley
Chris Cassidy
Becky Hames
And all of the Cluster 2 alumni!



Then and NOW

THE FUTURE OF ROBOTICS COMPETITIONS

by Tom Carroll

There have been interesting threads on the Seattle Robotics Society's (SRS), the Portland Area Robotics Society's (PARTS), and the Dallas Personal Robotics Group's (DPRG) websites lately concerning the state of experimental and home-brew robotics, and the many types of competitions available to amateur robot builders.

The SRS's Robothon, the PARTS PDXbot and BotFest, and the DPRG's Roborama are just a few of the many local contests and exhibitions around the country that both robot enthusiasts can show off what interests them and the general public can enjoy. One of the highest anticipated events each year is RoboGames, held in the San Francisco Bay area. It is the largest and most attended of all robot events, including industrial robotics. **Figure 1** is a group photo of some of the attendees and winners at the 2006 RoboGames. The 2010 event will be held in San Mateo, CA on April 23-25.

Certainly, everyone has their own ideas of what is most important in experimental robotics,

but there are clear paths of interest that always seem to emerge. I took some time at Robothon in Seattle Center this past October to talk with a few robot builders. Some people felt that interest in robot building was falling off whereas others felt that the availability of low-cost, ready-built robots gave enthusiasts an easy way to get into the hobby of robotics and the numbers of interested people were actually growing.

Everyone seemed to agree that the sluggish economy and tight discretionary money available to home-brew builders is changing people's spending habits. The print media is feeling the crunch, and magazines that keep us abreast of the latest information on experimental robot building

FIGURE 1.
The 2006
RoboGames at
Fort Mason
Center.





FIGURE 2.
The Quester
Micromouse,
courtesy of
David Buckley.

— this one included — certainly need our continued support. All of the smaller and most of the larger robot companies that advertise here in *SERVO* are struggling to stay in business to keep us supplied with the motors, servos, sensors, drivers, structural components, and robot kits that we need for our projects.

There is an upward turn in the economy as we start the new year, but we have a long way to go to achieve the stability that we all desire.

Is There Less Interest in Robot Conferences and Shows?

Many people noticed the drop in attendance and participation at the Seattle Robotics Society's Robothon this year and are wondering why. Is it the economy in the doldrums? Is it the large amount of unemployment? Are people shifting from technical hobbies to other interests? Or, could it just be the lower cost of finished robot components and systems that has driven people away from the 'build it yourself' attitude. Well, it's probably a bit of everything.

Some have said "Who needs contests? I build robots because I enjoy seeing my finished creation. My 'bots don't bash others to bits; they just make me happy." Quite a few people have expressed opinions about how the 'old tried and true' contests such as maze solving, micromouse contests, line following, Sumo, combat robots, and even RoboMagellan are getting a bit 'old in the tooth' and need some new life. FIRST, FRC, VEX, Robocup, BEST Robotics, and other 'country-wide' contests are still popular, and technical hobby exhibitions such as the iHobby Expos are getting a larger percentage of robotics enthusiasts. It seems to be the local variety held by individual robot clubs that are not as popular and are having fewer entrants than in the past.

The Micromouse

Let's examine a few of the 'staples' of contests that interest people the most. It is at these contests that builders get to show off their expertise in programming and basic robot design.

Foremost is probably the Micromouse contests. These were first envisioned by IEEE in 1977 and the first competition was held in New York in 1979. This autonomous robot contest has served as a model for many other events.

A micromouse is a totally autonomous robot that traverses and solves a 16 x 16 maze that is composed of 18 cm squares. The 'mouse' cannot crawl over or damage the maze walls and can be any height (though most winning mice are quite small with a low mass and CG to cover the course as quickly as possible). Entrants cannot see the maze (and program a solution ahead of time) until the start of a competition. At the start of a run, the mouse goes from a destination square on the perimeter of the maze to a destination square at the center. The quickest run time determines the winner. **Figure 2** is an early 1981 mouse and **Figure 3** is a more modern mouse from Japan.

Earlier mice used a wall following technique that would always result in a successful win. However, in one contest, a purely mechanical (non-intelligent) mouse won the competition, so now wall followers are banned. These events are a good display for the general public as something exciting and relatively easy to build. Go to ieee.org to learn more.

Other Robot Contests

Another popular contest is the line maze contest. **Figure 4** shows the line maze from the SRS's 2009 Robothon. Notice that there are 'tricky' parts to solve, such as endless loops and dead ends. It is basically a black line that the robot follows from the "T" to the black dot. The robot must fit within a six inch cube and be completely autonomous. Variations of this contest have included walled mazes similar to the Micromouse, simple line following, and obstacle traversing. Robots are custom-built from scratch or builders use LEGO or Vex kits and parts.

Robot versus robot contests include BattleBots, the Robot Fighting League, and similar variations, with 15 weight classes from 150 grams to almost 400 pounds, with and every type of weapon you can imagine. Sumo robot contests come in various weight sizes and classes as well. Both of these types of contests have radio control and autonomous varieties. They each draw a different type of crowd from the simple, single robots trying to solve a maze or navigation problem. Groups are now striving to vary the rules and classes to allow different competitions to draw more spectators and interest. SparkFun Electronics has just announced a new type of contest called RoboJoust that pits two combative

robots at opposite ends of a curved pathway, supported several feet in the air (see **Figure 5**). The winning robot navigates the curved path without slipping off and shoves the loser off the path and onto the floor. See details at sparkfun.com.

Robo-Magellan

The SRS Robo-Magellan contest was first envisioned by the Seattle Robotics Society and the first competition was held in 2004. Robo-Magellan has proven to be popular around the country. As the SRS rules state: "Robo-Magellan is a robotics competition emphasizing autonomous navigation and obstacle avoidance over varied, outdoor terrain. Robots have three opportunities to navigate from a starting point to an ending point, and are scored on the time required to complete the course with opportunities to lower the score based on contacting intermediate points."

A robot cannot weigh more than 50 pounds and must fit within a four foot cube. The robot must be totally autonomous, relying upon GPS coordinates, as well as visual navigation methods. Extra points are given when orange cones are detected and touched in the midst of the course.

Contestants are given the course coordinates 30 minutes before the beginning of the competition, and handheld GPS measurements and course leg lengths can be taken and programmed into the robot beforehand. **Figure 6** shows Mark Curry and Nomad, the winner of the 2009 SRS Robothon contest at Seattle Center. Many visitors at the huge Seattle Center were not aware of the robot show inside, and were surprised when occasionally one of the robots would start following someone wearing an orange jacket. This is the color of the cones the robots are programmed to find and touch. One woman with a robot following her in her orange dress commented, "I think it loves me." You can find Robo-Magellan details at robothon.org.

The New Future of Hobby Robotics

One does not need to be an expert or have extreme interest in all fields that encompass the science of robotics. Mechanics, electronics, and computer programming are the three main areas, and nobody is the best in all three. Add in sensor technology, vision, speech

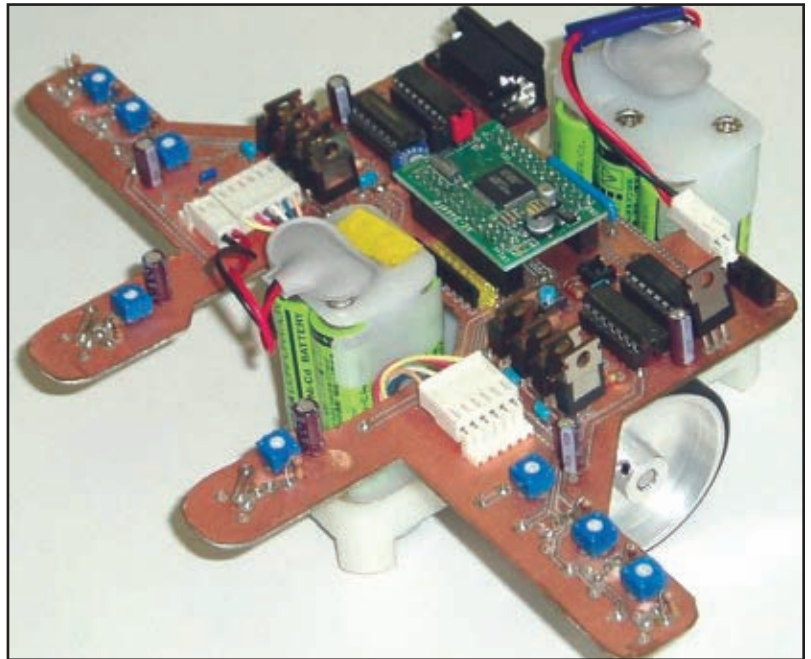


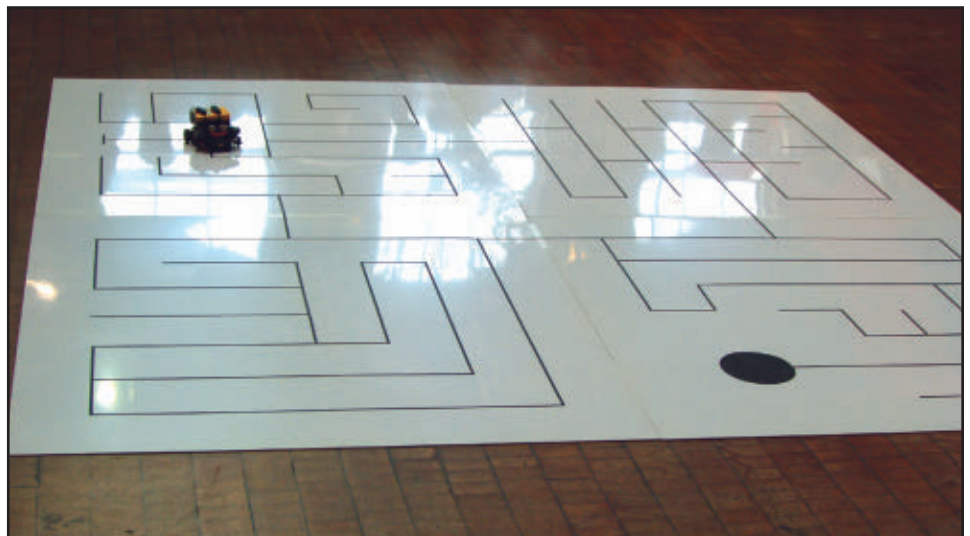
FIGURE 3.
A modern
Japanese
micromouse.

recognition and generation, navigation, chemical sensing, and appendages with end effectors (claws), and there are more aspects of today's robot building that can allow everyone to participate and feel like an expert.

Newer Robot Competitions

Many of the discussions from the different robot groups has been to try and determine what are the most interesting events and what new ones would entice others to join a robotics group and design and build experimental robots. One suggestion was a telepresence competition. In previous years, cameras were of the monochrome vidicon variety or expensive CCD devices out of the reach of most experimenters. Nowadays, fairly

FIGURE 4. The
robot solving
line maze at
Robothon.



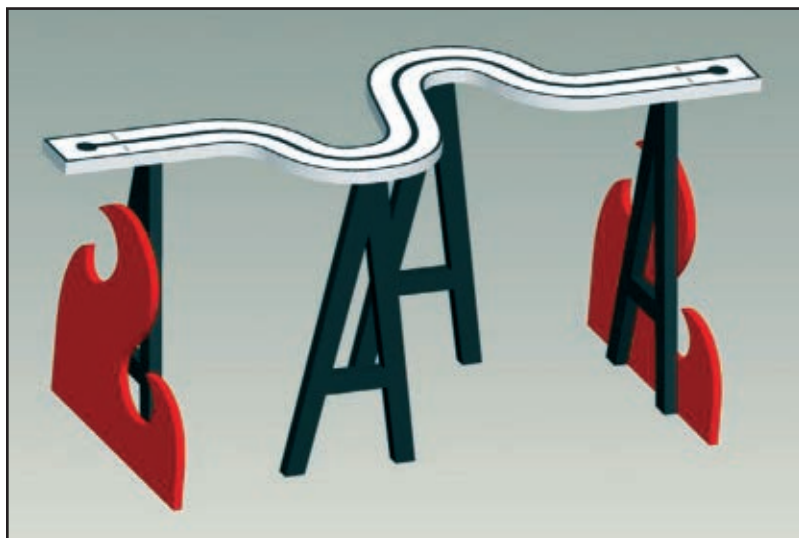


FIGURE 5.
SparkFun's
Robojoust
table.

high resolution cameras with a built-in transmitter are fairly cheap.

Telepresence competitions can be designed around a scientific remotely-controlled robot exploring a cave, underwater location, the moon, or even Mars. Control delays can be built in to simulate the moon's three second delay or Mars' delay of several minutes. Scoring can be based on speed, retrieval of 'moon rocks,' location of water — all done by remote control from a location hidden from the competitor's view.

Observers can watch the actual rovers operate. Both R/C and autonomous functions can be used, and the more sensors used, the easier the task. Just as Robo-Magellan was based on the DARPA Grand Challenge, scaled-down robotic competitions can mimic present-day scientific uses and allow home experimenters to build smaller

robots to explore undersea, simulated space conditions, or laboratory environments.

Newer Robot Designs

Basic home-built robot designs have changed very little over the past three decades. Virtually all experimenter's wheeled robots employ differential steering. The reason for this is simple: If you apply the same power to each wheel and count the wheel turns by an encoder or use a stepper motor, the robot will go straight. Opposite polarity to the wheels will cause the robot to rotate on its axis, and variations will allow the robot to turn. This works fine if there is no wheel slippage.

The Ackermann design is a bit more complex for beginners as it requires building interconnected steering mechanisms for the front (or back) wheels, and the programming requires input to a steering motor and to the main wheel drive motor(s). If designed correctly, the resulting steering can be much more accurate than differential steering but a lot of builders find the programming a bit beyond their capabilities. Several robots are now using off-road R/C car bases with Ackermann steering, and adding a microcontroller and sensor suite to build a great robot.

Experimenter-level robots used switches as feelers for years, but now have started using IR/visible light beams to detect obstacles and walls. Maxsonar, Devantech, and Parallax make nice and inexpensive ultrasonic sensors to detect objects. However, the old Polaroid electrostatic sonars with their narrow beam are still a favorite with many builders. The (far) more expensive but vastly superior laser range finders made by Sick, Hokuyo, and other high end suppliers are finding their way onto a few more sophisticated robots. The selection of sensors of the above might be dependent upon what type of background the robot is operating with, such as grass, carpet, or a hard floor. Balancing robots utilize accelerometers and gyros for sensors. You can get a combination of both with SparkFun's 6DOF Razor that has gyros and accelerometers in all three axes, available on a small printed circuit board.

Intelligent cameras such as the CMU camera can be indispensable as a main sensor for more complex robot designs. Many people are looking to go beyond a simple differential wheel robot that can follow a maze and develop a robot that can operate in the cluttered environment of a home. Dave Shinsel is shown in **Figure 7** demonstrating Loki to SRS's Cathy Saxton at the 2009 Robothon. Using simple web cams and deriving visual intelligence through the use of an

FIGURE 6.
Mark Curry
and Nomad at
Robothon's
Robo-Magellan.



onboard laptop computer, this robot can steer through doorways and avoid other obstacles. Loki uses a long-range IR sensor. Now equipped with two functional arms and claws, Loki can pick up and manipulate objects, controlled by his vision just like we humans do. Other experimenters are working on robots that monitor their homes, retrieve a Pepsi from the fridge, or even walk their dogs, all with multiple sensors in a suite interconnected to one or more microcontrollers.

Walking robots have always been interesting to build, and many experimenters start out with hexapod walkers using model airplane servos for single, two, or three axis legs. Quadrapods are popular, but it is the humanoid biped robots that have sparked a turning point in robot design. All of the major servo manufacturers brought forth their own versions of a humanoid – sometimes several from one manufacturer. With 18 or more servos per robot, it was hard to keep many humanoids below \$1,000 – a tough purchase for many experimenters.

Now we're starting to see some sophisticated underwater robotic vehicles – both the autonomous variety and ROVs – being built by experimenters. Autonomous helicopters built from modified R/C vehicles and even multi-rotor flyers are taking to the air with cameras and GPS guidance systems. Low cost microcontroller systems such as the Arduino platform based on the Atmel AVR controller (and others) are becoming the system of choice, as well as the old standbys like the BASIC Stamp, Propeller, 68HC11-12, and many PIC variations.

The Philosophy Behind Robot Building

Some people start out to build a robot to compete in a contest at their local clubs or maybe at one of the larger events. When the event is over and they bring their creation home, they begin to wonder "What can this robot of mine do in the real world? What improvements can I make to have it function in my house?" They may have solved the GPS navigation for Robo-Magellan, only to find out that they'll need a whole new navigation system to work within their home environment.

As an incentive to build a robot, long-time *SERVO* columnist, Pete Miles, said, "Most of us build robots because this is what we like to do." A builder may start out learning various mechanical, electrical, and programming techniques with a

LEGO or other kit-built robot. At that point, they may decide to enter one of the more basic contests to see how their machine compares with others. If the builder has some mechanical skills, they may decide to 'cut metal' and design their own robot. If their expertise is in electrical control systems or interfacing microcontrollers with motor driver and sensors, they may take a route similar to what Dave Shinsel did when he built his robot Loki and scrounge everyday items that "look like robot parts."

Final Thoughts

Experimental and hobby robotics is evolving. The learned skills are valuable to today's technology. There is no technology that encompasses more engineering disciplines than robotics, and nationwide robotics competitions are convincing many students that this course of study will best prepare them for future careers in engineering and the sciences. What is the future of robotics? The future is bright but there are some significant changes in the works. **SV**

FIGURE 7.
David Shinsel
showing Loki
to Cathy
Saxton.



Tom Carroll can be reached at
TWCarrall@aol.com.

ROBO-LINKS

BiPOM Electronics, Inc. Your One Source for μ Controller Systems
ARM, AVR, PIC, 8051, 6801, STAMP and others

μ Controller Boards
Peripherals
Development Tools
Emulators
Programmers
Robotics

Sensors
Dev. Training Kits
Instruments
Displays

GadgetPC ARM9 USB
Computer with Linux

www.bipom.com

WebCat+
Embedded Web Server

GPS MADE SIMPLE™

Linx Technologies

LinxTechnologies.com

IMAGES Scientific Instruments

SCIENCE,
ROBOTICS &
ELECTRONICS

Microcontrollers
Servo Control & Motors
Artificial Vision
Speech Recognition

www.imagesco.com

Tel: (718) 966-3694 Fax: (718) 966-3695

RobotShop.com

THE ORIGINAL SINCE 1994

PCB-POOL

Beta LAYOUT

- Low Cost PCB prototypes
- Free laser SMT stencil with all Proto orders

WWW.PCB-POOL.COM

Pololu

Robotics & Electronics

3095 E. Patrick Ln. #12, Las Vegas, NV 89120

www.POLOLU.com 1-877-7-POLOLU

Motor, servo, and robot controllers
Sensors

Robots, motors, wheels, and more!

\$29.95 MaxSonar-EZ1 (MSRP)

High Performance
Ultrasonic Range Finder

- serial, analog voltage & pulse width outputs
- lowest power - 2mA
- narrow beam
- very easy to use!

www.maxbotix.com

For the finest in robots, parts, and services, go to www.servomagazine.com and click on Robo-Links.

The most comprehensive resource for news and links

THE ROBOT REPORT

TRACKING THE BUSINESS OF ROBOTICS

www.TheRobotReport.com

ALL ELECTRONICS CORPORATION

Electronic Parts & Supplies
Since 1967

5 CD-ROMs & Hat Special

Only \$ 109.00 (includes shipping)!

www.servomagazine.com

ADVERTISER INDEX

All Electronics Corp.51, 82	Images Co.82	Robotis2
AP Circuits45	Integrated Ideas & Tech65	RobotShop, Inc3, 82
BaneBots32	Linx Technologies82	Solarbotics/HVW11
BiPOM Electronics82	Lynxmotion, Inc.83	Techno-Swap-Fest66
CrustCrawler37	Maxbotix82	The Robot Report82
DigilentBack Cover	PCB Pool59, 82	WeirdStuff Warehouse51
Fast Track Training Pubs11	Pololu Robotics & Electronics ..19, 82	
Galaxy Far East Corp.51	Robot Power45	

**To Advertise:
Call 951 371-8497**



The Lynxmotion Servo Erector Set Imagine it... Build it... Control it!

Featured Robot

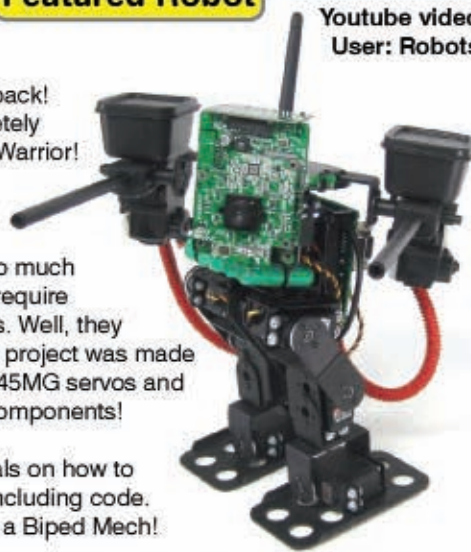
**Biped BRAT does
Mech Warfare!**

Introducing the Hunchback!
The world's first completely
functional biped Mech Warrior!

They said a BRAT
based Mech couldn't
be done. That it was too much
payload. That it would require
expensive digital servos. Well, they
were wrong! This robot project was made
from a BRAT with HS-645MG servos and
standard off the shelf components!

We have created tutorials on how to
make a Mech Warrior including code.
Now anyone can make a Biped Mech!

Youtube videos
User: Robots7



Biped Nick



Biped Pete



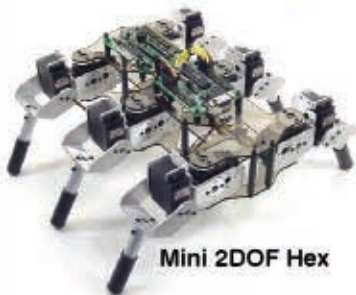
Biped Scout



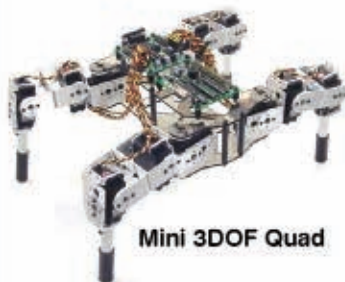
Biped 209



Walking Stick



Mini 2DOF Hex



Mini 3DOF Quad

**With our popular Servo Erector Set you can easily
build and control the robot of your dreams!**

Our interchangeable aluminum brackets, hubs,
and tubing make the ultimate in precision
mechanical assemblies possible. The images here
are just a sample of what can be built. The Bot
Board II and SSC-32 provide powerful control
options. Our Visual Sequencer program provides
powerful PC, Basic Atom, or BS2 based control.



Bot Board II - \$24.95
Carrier for Atom / Pro, BS2, etc.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
Buffered Speaker.
Sony PS2 game controller port.
3 Push button / LED interface.



SSC-32 - \$39.95
32 Channel Servo Controller.
Speed, Timed, or Group moves.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
TTL or RS-232 Serial Comms.
No better SSC value anywhere!

We also carry motors, wheels, hubs, batteries, chargers,
servos, sensors, RC radios, pillow blocks, hardware, etc!



Visit our huge website to see our complete line
of Aluminum and Lexan based robot kits,
electronics, and mechanical components.



CH3-R Hexapod



Biped BRATs

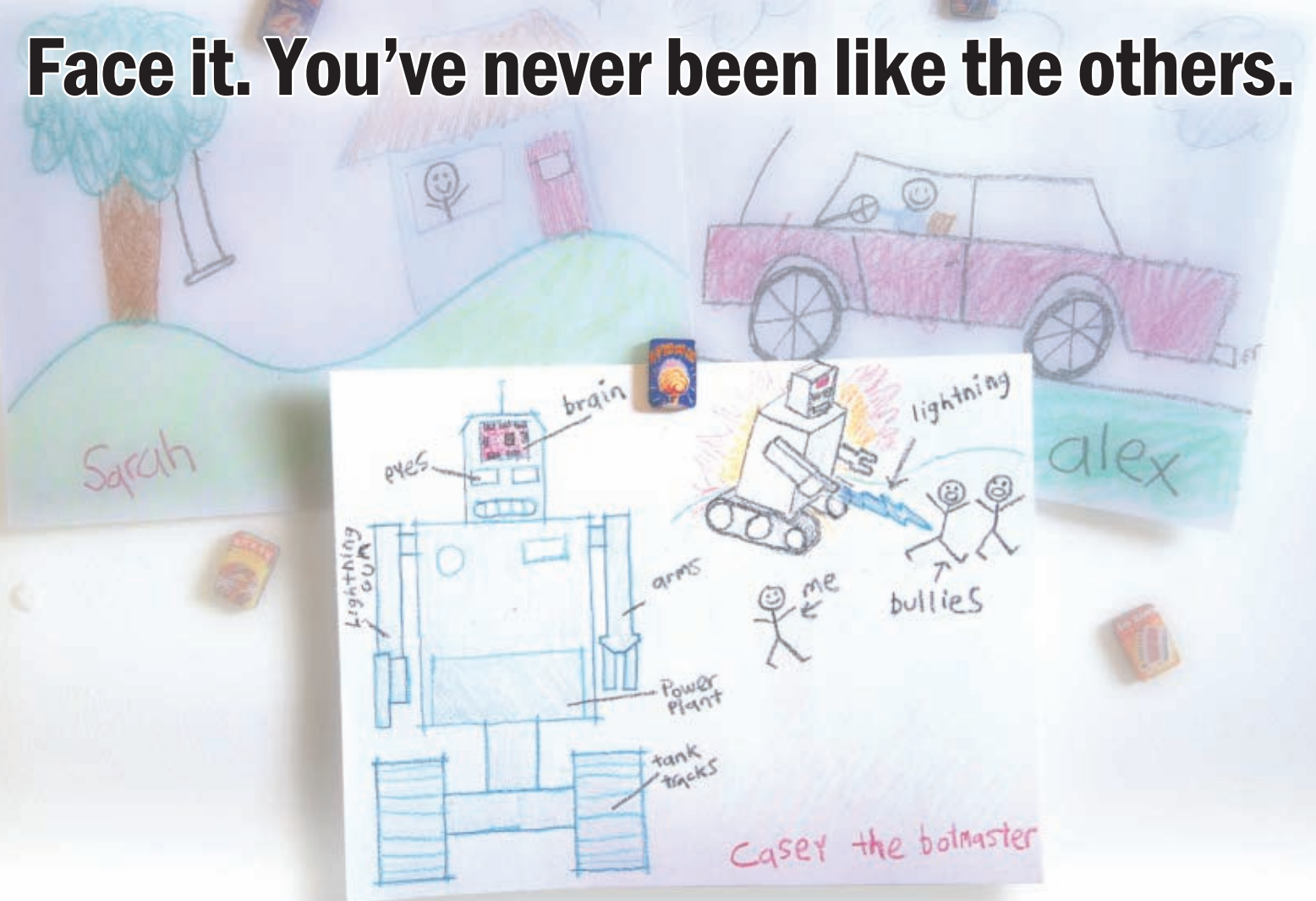


Phoenix



Images represent a fraction of what can be made! www.lynxmotion.com The SES now has 157 unique components!

Face it. You've never been like the others.



We know how you feel.

CEREBOTTM
32MX4

\$54

Enter the value code "servo2010a" on our website for this special introductory price

A powerful PIC32[®] based microcontroller board, ideal for robotic experimentation in C, C++, and Assembler.

- Microchip[®] PIC32MX460F512L
- 80 MHz 32-bit MIPS processor
- 512K Flash, 32K RAM
- Eight R/C servo connectors, two SPI ports, two I²C ports, two UARTs
- Nine 12-pin Pmod[™] Connectors
- Five 16-bit timers with 5 input capture and 5 PWM outputs
- 16 channel, 10-bit, 500kps A/D converter
- On-board USB Program / Debug Interface
- USB OTG Host / Device Capable
- USB Powered



Pmod[™]
Peripheral Modules

\$9.99 - \$29.99

Expand your designs

Pmods are a wide range of small, inexpensive & versatile I/O interface boards.



PmodJSTK
2-axis joystick
w/ 3 pushbuttons



PmodHBS
2A H-bridge
w/ feedback inputs



PmodENC
Rotary encoder module
w/ integral pushbutton



PmodMIC
Electret mic w/
compressor, & 12-bit A/D



PmodCLS
2-line character LCD
w/ serial interface



Digilent, Inc. is a leader in the design & manufacture of FPGA and microcontroller technologies. Our products can be found in over 100 countries and over 2000 universities worldwide. Visit our website for a wide assortment of these boards, as well as peripherals, reference designs, sample projects, tutorials, textbooks, and more.

www.digilentinc.com